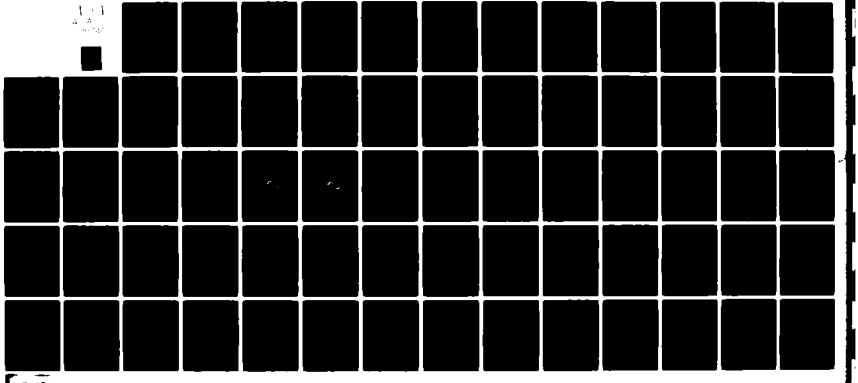
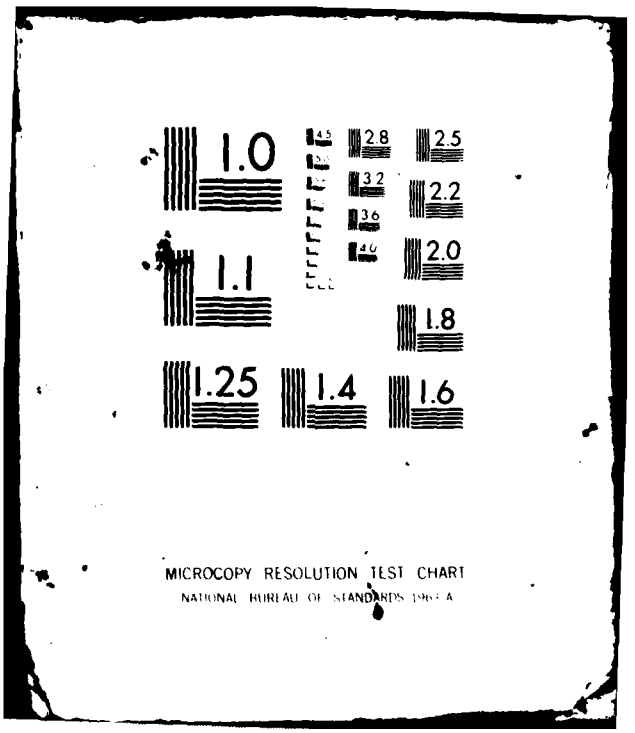
AD-A108 746 WISCONSIN UNIV-MADISON DEPT OF STATISTICS F/G 12/1
THE COMPUTATION OF LAPLACIAN SMOOTHING SPLINES WITH EXAMPLES.(U)
SEP 81 J G WENDELBERGER N00014-77-C-0675
UNCLASSIFIED TR-648 NL



END
DATE FILMED
1 82
DTIC


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963 A

DEPARTMENT OF STATISTICS

University of Wisconsin
1210 W. Dayton St.
Madison, WI 53706

TECHNICAL REPORT NO. 648

September 1981

THE COMPUTATION OF LAPLACIAN SMOOTHING SPLINES WITH EXAMPLES

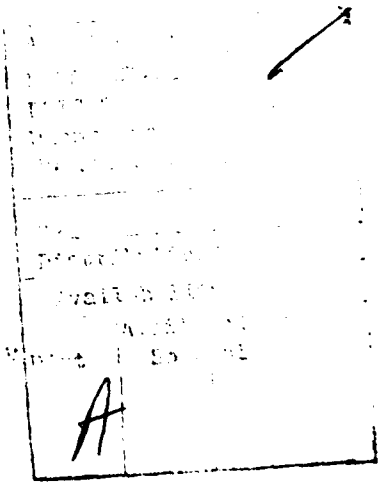
James G. Wendelberger

University of Wisconsin-Madison

This research was supported by NASA under Grant No. NAG5-128 and by the Office of Naval Research under Contract No. N00014-77-C-0675.

ABSTRACT

Laplacian Smoothing Splines (LSS) are presented as generalizations of graduation, cubic and thin plate splines. The method of generalized cross validation (GCV) to choose the smoothing parameter is described. GCV is used in the algorithm for the computation of LSS's. An outline of a computer program which implements this algorithm is presented along with a description of the use of the program. Examples in one, two and three dimensions demonstrate how to obtain estimates of function values with confidence intervals and estimates of first and second derivatives. Probability plots are used as a diagnostic tool to check for model inadequacy.



1. Motivation

A Laplacian smoothing spline (LSS) is a statistical tool used to model a smooth but otherwise unknown function. The fitted spline provides an analytic function which may be utilized to estimate derivatives, integrals or values of the underlying function. For data analysis purposes a graphical display of the fitted spline (or cross sections for multidimensional problems) often provides insight which might otherwise remain masked by the irregularly spaced, multidimensional and "noisy" data. The residuals, which are the observed values of the dependent variable minus the corresponding fitted spline values, may be utilized as an aid in model checking. A probability plot of the residuals provides a vehicle to detect possibly discrepant observations (outliers). With the above ideas as the eventual objective we first elucidate the functional form of the LSS and then describe an algorithm for its computation.

When someone mentions a line, cosine or an exponential we all have a visual image of "feel" for the function in question. Using the following example we hope to provide an intuitive feeling for an LSS.

In one dimension imagine a long, thin, perfectly rigid rod (a line) lying on a frictionless plane with coordinate axes (t,z). We represent this rod as a function of t, say g(t). Assume that we are given N points in the plane $\{(t,z):(t,z)=(t_i,z_i),\ i=1,\ldots,N\}$. The t_i are considered to be distinct and known without error. The z_i are measurements of a true but unknown function f evaluated at t_i plus some "noise" e_i. The e_i are independent random variables, each having mean zero and finite variance.

With the previous setup imagine that an ideal spring is attached to data point (t_i,z_i) and to the rod $(t_i,g(t_i))$ for each i, i=1,...,N. This fixes the springs to remain parallel to the ordinate axis. What position will the rod g(t) assume?

Physics provides a means to answer this question. The rod will assume the position which minimizes the energy of the springs. The energy of an ideal spring is equal to some positive constant k_i (called the spring constant) times the square of the length it is stretched. Thus the cumulative energy of the N springs is

$$\sum_{i=1}^{N} k_i(z_i - g(t_i))^2 .$$

This is minimized when g is the least squares line (provided we restrict g to be *rigid*) therefore the least squares line is the position the rod will assume if $k_i = k_o$, i=1,...,N, k_o some constant. If the k_i are not all equal then the rod will assume the position of the weighted least squares line. Notice that this spring idea provides an intuitive explanation for minimizing the residual sum of squares in regression.

The situation is analogous in two dimensions: a thin plate of infinite rigidity (not bendable) would assume the position of the least squares plane. The situation in three dimensions, although not as easy to visualize, is analogous. There are further restrictions on the t_i which are rigorously given in (2.6).

We have thus far assumed that the rod is rigid. This is not necessary and may not be a good representation of the physical phenomenon under

consideration. So we relax the rigidity assumption and assume that the rod is flexible. If zero energy were required to flex the rod then the minimum energy position which the rod would assume is that of a function of interpolation. Since the residuals are zero, this configuration has zero energy and thus is a minimum. By this explanation it is readily seen that the function thus obtained is not unique. This anomaly will be alleviated by requiring energy to flex the rod.

Consider the more realistic case where the rod is flexible and takes energy to flex. The spring of a diving board is testimony to this. Note that the bending energy of a rod is $(\rho/\sigma^2)J_2(g)$, where ρ/σ^2 is a constant and

$$J_2(g) = \int_{-\infty}^{\infty} [g^{(2)}(x)]^2 dx \ . \tag{1.1}$$

Therefore the bending energy is proportional to curvature which may be measured as $J_2(g)$ in (1.1).

To find the position which the rod will assume under these conditions is equivalent to finding the function g which will minimize the total energy of the system

$$\sum_{i=1}^{N} k_i(z_i - g(t_i))^2 + (\rho/\sigma^2)\, J_2(g) \tag{1.2}$$

or equivalently the minimizer of

$$(1/N) \sum_{i=1}^{N} \sigma^2 k_i(z_i - g(t_i))^2 + (\rho/N)J_2(g) \ . \tag{1.3}$$

The function from a certain class of functions, X, which minimizes (1.3) can be shown to be a piecewise cubic spline. The function space X is

rigorously defined in Wahba and Wendelberger (1980). Here X should be thought of as a space of smooth functions which map R^d into R^1. There is much literature about cubic splines in one dimension. To this author's knowledge the earliest work on LSS's is that of Schoenberg (1964); other important work on splines is given in Craven and Wahba (1979), Duchon (1976), Prenter (1975), and Reinsch (1967).

The one dimensional case generalizes to two dimensions. In two dimensions the splines are called thin plate splines because of the analogy of minimizing the energy of a thin plate of infinite extent. The earliest suggested application of thin plate smoothing splines seems to have been by Harder and Desmarais (1972). They suggested that spring forces may be applied at the points of interpolation. This inspired the spring analogy given here. This spring concept is equivalent to LSS's in either one or two dimensions (with m=2 in (2.1)). Much recent work on LSS's has been done by Wahba (see Wahba (1979) and the references cited there).

In two dimensions $J_2(g)$ becomes

$$J_2(g) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\sum_{\nu=0}^{2}\binom{2}{\nu}\left[\frac{\partial^2 g(x_1,x_2)}{\partial x_1^{\nu}\,\partial x_2^{2-\nu}}\right]^2 dx_1\,dx_2 \ . \tag{1.4}$$

$J_2(g)$ is proportional to the bending energy of a thin plate (under simplifying assumptions); for details see Meinguet (1979). However, in two dimensions the solution is no longer a piecewise cubic but rather takes the form

$$g(\underset{\sim}{t}) = \sum_{i=1}^{N} c_i'\tau_i^2\ln(\tau_i) + d_0 + d_1x_1 + d_2x_2 \ , \tag{1.5}$$

where τ_i is the Euclidean distance between $\underset{\sim}{t}$ and $\underset{\sim}{t}_i$, that is $\tau_i^2 = |\underset{\sim}{t}-\underset{\sim}{t}_i|^2$

$= (t_{i1} - x_1)^2 + (t_{i2} - x_2)^2$; t_{ij} is the j^{th} component of $\underset{\sim}{t}_i$, $j=1,2$, $\underset{\sim}{t} = (x_1,x_2)$; c_i' and d_ν are constants, $i=1,\ldots,N$, $\nu=0,1,2$.

To aid in understanding (1.5) the function $\tau_0^2 \ln(\tau_0)$ is plotted in Figure 1.1 for $\underset{\sim}{t}_0 = (0,0)$ and $x_2 = 0$. Rotation of this function around the ordinate axis and centering at the point $\underset{\sim}{t}_i$ will produce the radially symmetric function $\tau_i^2 \ln(\tau_i)$. Using (1.5) an LSS is seen to be composed of a linear combination of these radially symmetric functions plus a plane. The plane has zero bending energy but generally does have nonzero spring energy. Linear combinations of the radially symmetric functions can be forced to interpolate the points and hence may have zero spring energy but generally have nonzero bending energy. This tradeoff between bending and spring energy, or smoothness and infidelity to the data (terminology of Wahba (1979)), leads one to consider the minimization problem of Section 2 as a generalization of these ideas. The one and two dimension examples with m=2 are special cases of this generalization.

We see that the motivation for one and two dimensional LSS's is quite simple (at least for m=2). Attach springs to the data points, constrain them to lie perpendicular to the independent variable space R^d, then let the curve or surface conform by simple bending to the minimum energy configuration.

The Laplacian smoothing spline was suggested by Duchon (1976) as a multidimensional generalization of the thin plate (or "plaques minces"), d=2, interpolating spline. An LSS is also a multivariate generalization of the one dimensional, d=1, "graduation" spline of Schoenberg (1964). Furthermore, the "graduation" spline is a generalization of the familiar cubic smoothing

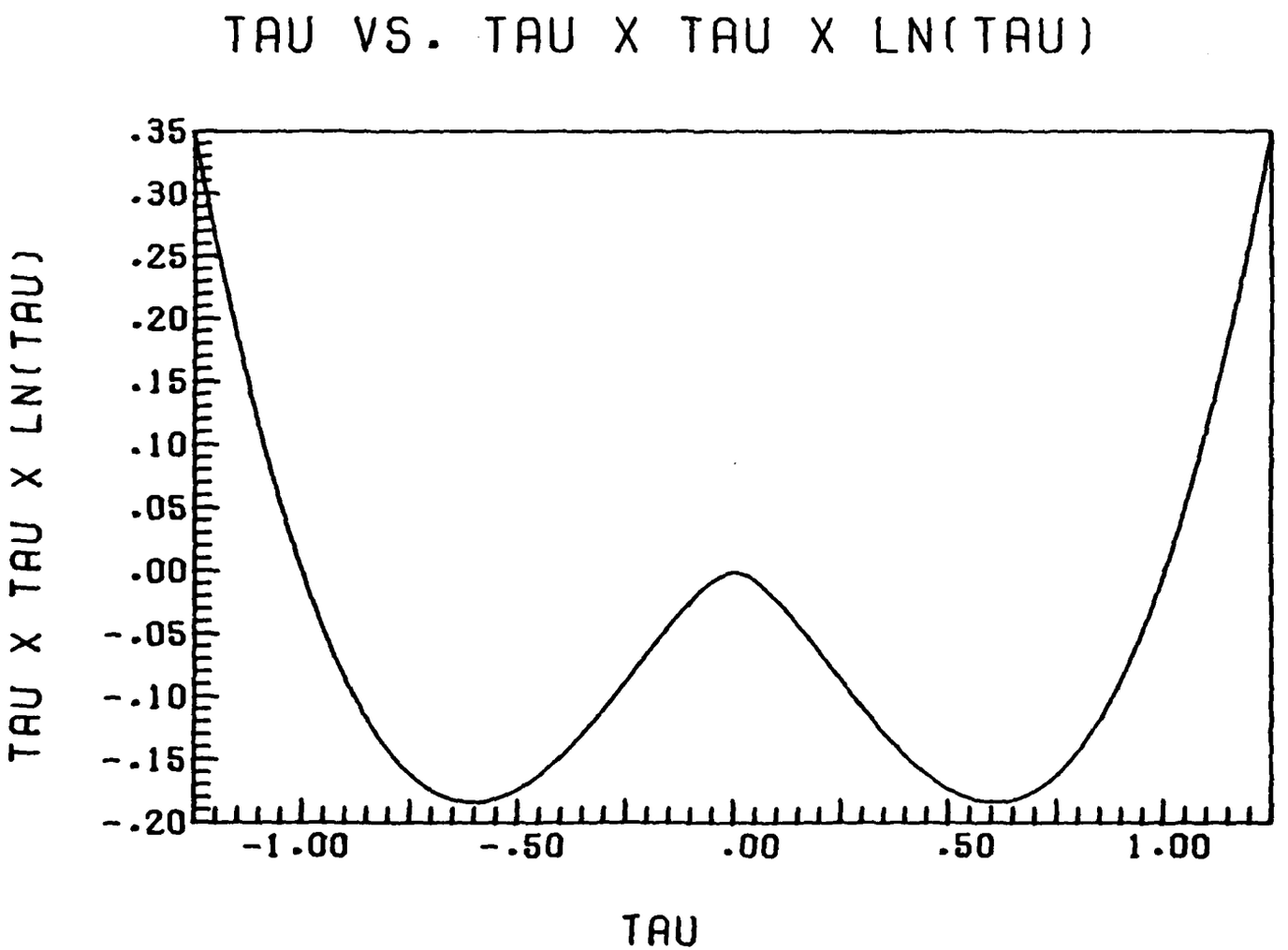


Figure 1.1: τ_o vs. $\tau_o^2 \ln \tau_o$.

spline. The terminology "Laplacian smoothing spline" was suggested by Professor I. J. Schoenberg. An explanation for using the term "Laplacian" is given in Wahba (1979).

2. Characterization

Let $z_i = f(\underset{\sim}{t}_i) + e_i$, $i=1,\ldots,N$. The $\underset{\sim}{t}_i \epsilon R^d$ are known exactly. We assume that the function f is smooth but otherwise unknown. By smooth it is meant that the function is well approximated by a function $g \epsilon X$; X is rigorously defined in Wahba and Wendelberger (1980). X may be thought of as a space of functions which approximate well a large class of functions of which f is a member. The e_i are independent, zero mean and finite variance random variables with variance-covariance matrix $\sigma^2 D_\sigma^2 = \sigma^2 \mathrm{diag}(\sigma_1^2,\ldots,\sigma_N^2)$. Here σ^2 is an unknown constant. For example, if we know that all the variances are equal then we may take $1.0 = \sigma_1^2 = \ldots = \sigma_N^2$ in what follows. The σ_i^2 used here are inversely proportional to the k_i of Section 1, that is, $k_i = (\sigma\sigma_i)^{-2}$. The σ_i^2 may be thought of as relative weights of the measurement errors e_i. The z_i are observed dependent variables in R^1 and the corresponding $\underset{\sim}{t}_i$ are independent variables in R^d, $i=1,\ldots,N$.

A Laplacian smoothing spline is the function g which is the solution to the problem.

Find $g \epsilon X$, X a suitable function space, such that

$$N^{-1}||D_\sigma^{-1}(\underset{\sim}{z}-\underset{\sim}{g})||^2 + (\rho/N)J_m(g) \tag{2.1}$$

attains its minimum. Here define

$$\underset{\sim}{z} = (z_1,\ldots,z_N)^T,\ \underset{\sim}{g} = (g_1,\ldots,g_N)^T,\ g_i = g(\underset{\sim}{t}_i),\ ||D_\sigma^{-1}(\underset{\sim}{z}-\underset{\sim}{g})||^2$$
$$= (\underset{\sim}{z}-\underset{\sim}{g})^T D_\sigma^{-2}(\underset{\sim}{z}-\underset{\sim}{g}),\ D_\sigma^{-1} = \mathrm{diag}(\sigma_1^{-1},\ldots,\sigma_N^{-1}),$$

where superscript T means transpose throughout. Also,

$$J_m(g) = \sum_{\nu=1}^{M'} \frac{m!}{\alpha_{1,\nu}!,\ldots,\alpha_{d,\nu}!} \int_{-\infty}^{\infty}\cdots\int_{-\infty}^{\infty} \left[\frac{\partial^m g(\underset{\sim}{t})}{\partial x_1^{\alpha_{1,\nu}},\ldots,\partial x_d^{\alpha_{d,\nu}}}\right]^2 dx_1,\ldots dx_d; \tag{2.2}$$

$\underset{\sim}{t} = (x_1,\ldots,x_d)^T$; $M' = \binom{m+d-1}{d-1}$; the $\alpha_{1,\nu}, \ldots, \alpha_{d,\nu}$ are the M' unique combinations of $\{0,1,\ldots,m\}$ such that $\alpha_{1,\nu}+\ldots+\alpha_{d,\nu} = m$.

In the case presented earlier with $d=2$ and $m=2$ we have $M'=3$ and $(\alpha_{1,\nu},\alpha_{2,\nu})$ takes on the M' unique values (1,1), (2,0) and (0,2). In this case (2.2) reduces to (1.4).

The solution to the minimization problem is unique and given in (2.3).

$$g(\underset{\sim}{t}) = \sum_{i=1}^{N} c_i \theta_{m,d} \tau_i^{2m-d} (\ln \tau_i) I_e(d) + \sum_{\nu=1}^{M} d_\nu \phi_\nu(\underset{\sim}{t}) , \tag{2.3}$$

where I_e is the indicator function of even integers, that is $I_e(d)=1$, for d even and $I_e(d)=0$, for d odd;

$$\theta_{m,d} = \begin{cases} (-1)^{d/2+1+m}/(2^{2m-1}\pi^{d/2}(m-1)!(m-d/2)!), & d \text{ even} \\ \Gamma(d/2-m)/(2^{2m}\pi^{d/2}(m-1)!), & d \text{ odd} \end{cases} \tag{2.4}$$

and ϕ_ν are the polynomials of total degree less than m,

$$\phi_\nu(\underset{\sim}{t}) = \phi_\nu(x_1,\ldots,x_d) = x_1^{p_{1\nu}} \ldots x_d^{p_{d\nu}} . \tag{2.5}$$

Here the ϕ_ν are unique; $p_{i\nu} \geq 0$, $i=1,\ldots,d$ and $p_{1\nu}+\ldots+p_{d\nu} < m$, $\nu=1,\ldots,M$, $M = \binom{m+d-1}{d}$. Define the M by d matrix P to have $i\nu^{th}$ element $p_{i\nu}$. Also, *$2m-d > 0$ and (2.6) holds.*

$$\sum_{\nu=1}^{M} a_\nu \phi_\nu(\underset{\sim}{t}_i) = 0, \; i=1,\ldots,N \text{ implies } a_\nu = 0, \; \nu=1,\ldots,M . \tag{2.6}$$

(Condition (2.6) requires that the matrix T_σ of Section 5 step (ii) be of rank M.) $\underset{\sim}{c} = (c_1,\ldots,c_N)^T$ and $\underset{\sim}{d} = (d_1,\ldots,d_N)^T$ are obtained by solving the linear system

$$(K + \rho\sigma^2 D_\sigma^2)\underset{\sim}{c} + T\underset{\sim}{d} = \underset{\sim}{z} \tag{2.7}$$

and

$$T^T c = 0 . \tag{2.8}$$

In (2.7) K is the N by N matrix with ij^{th} element $\theta_{m,d}\tau_{ij}^{2m-d}(\ln(\tau_{ij}))I_e(d)$. In (2.7) and (2.8) T is the N by M matrix with $i\nu^{th}$ element $\phi_\nu(t_i)$. In (2.7) D_σ^2 is the N by N diagnal matrix with ii^{th} entry σ_i^2. σ^2 is an unknown proportionality constant which along with ρ is absorbed into λ using $N\lambda = \rho\sigma^2$ to yield (2.9) from (2.7).

$$(K + N\lambda D_\sigma^2)c + Td = z \tag{2.9}$$

The approach of Harder and Desmarais (1972) provides us with a physical interpretation of the parameters at least in the d=2 case. $\rho=N\lambda\sigma^{-2}$ is the plate "rigidity" which is a constant. The value of ρ depends on the material and the thickness of the plate. The spring constant k_j is equal to the reciprocal of the variance or $(\sigma\sigma_j)^{-2}$. The "load" at the j^{th} point is $P_j = \rho c_j = (\sigma\sigma_j)^{-2}r_j = k_j r_j$, where r_j is the unnormalized or unscaled residual at that point; i.e., $r_j = z_j - g(t_j)$, j=1,...,N or $r = z - Kc - Td$.

For a discussion of a more general problem and the derivation of the solution the reader is referred to Wahba and Wendelberger (1980). We note here that if the e_i are not independent but instead have positive definite covariance matrix proportional to Σ then D_σ^2 and D_σ^{-1} are everywhere replaced by Σ and the symmetric inverse square root $\Sigma^{-1/2}$ to obtain the solution.

To this point we have assumed knowledge of the smoothness parameter λ. However it is generally unknown. Before describing a method to dynamically choose λ from the data at hand we provide an example to exhibit its influence on the LSS.

3. Example 1 — Variation of the LSS with λ, d=1.

A company which makes and repairs small computers wants to forecast the number of service engineers that it will require over the next few years. To do this requires, among other things, knowledge of the length of a service call. The length of a call is a function of the number of components within the computer which must be repaired or replaced. The information in Table 3.1 was collected on 24 service calls; the data are from Chatterjee and Price (1977). We would like to fit a spline to the data in order to forecast the length of a service call.

We fit a spline to the data using the algorithm given in Section 5. The smoothness parameter, λ, is dynamically chosen from the data using the method of generalized cross validation (GCV). By showing the influence of λ on the LSS of this example we hope to provide a clearer understanding of the role of GCV in choosing the smoothness parameter. The results of the following sections will be easier to understand with this example in mind. Exactly what the GCV choice of λ is will be presented in Section 4.

Figure 3.1 shows a plot of the data and the corresponding spline for five different values of λ. Because there are only 24 observations of which only 17 have unique independent variables we should not be surprised if the GCV estimate (to be described in Section 4) of λ, which is a large sample result, does not perform well. The confidence intervals are calculated using method of Wahba (1981); the formula used for their computation is given in Example 2 of Section 6.

TABLE 3.1

EXAMPLE 1 - REPAIR TIMES

Length of Calls (Minutes)	Units Repaired (Number)
23	1
29	2
49	3
64	4
74	4
87	5
96	6
97	6
109	7
119	8
149	9
145	9
154	10
166	10
162	11
174	11
180	12
176	12
179	14
193	16
193	17
195	18
198	18
205	20

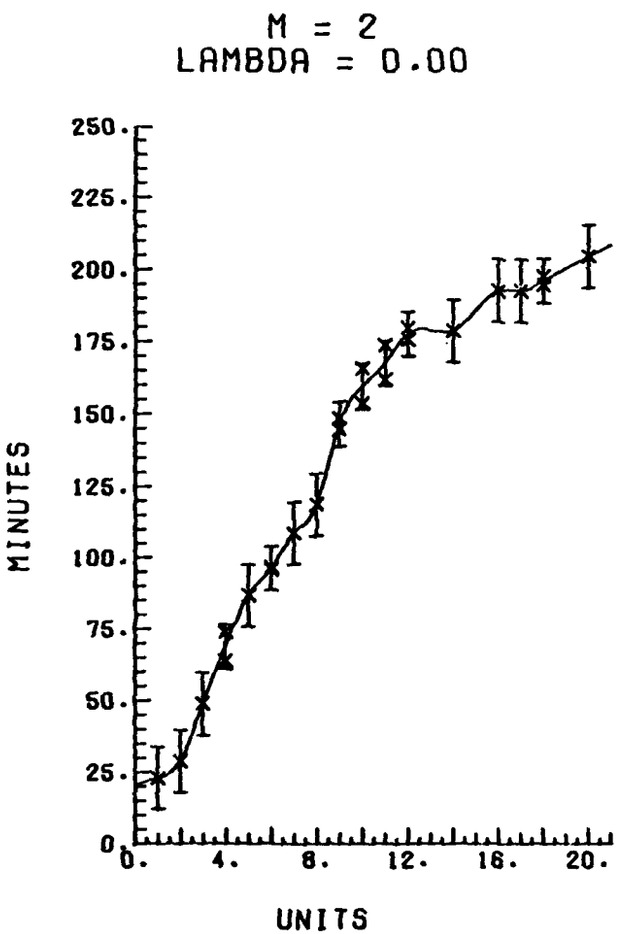


Figure 3.1a: Example 1 with $\lambda = 0.00$

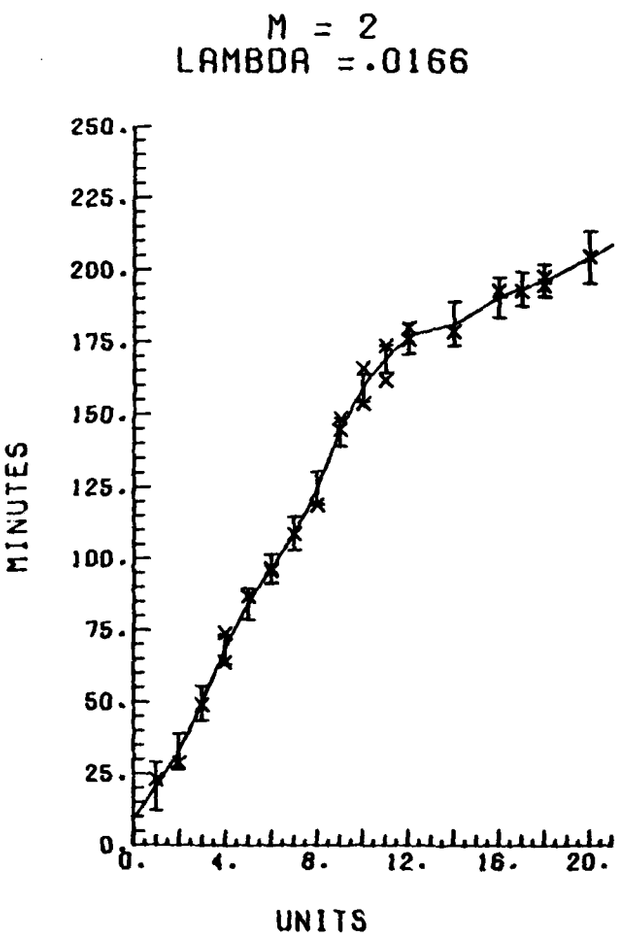


Figure 3.1b: Example 1 with $\lambda = 0.0166$

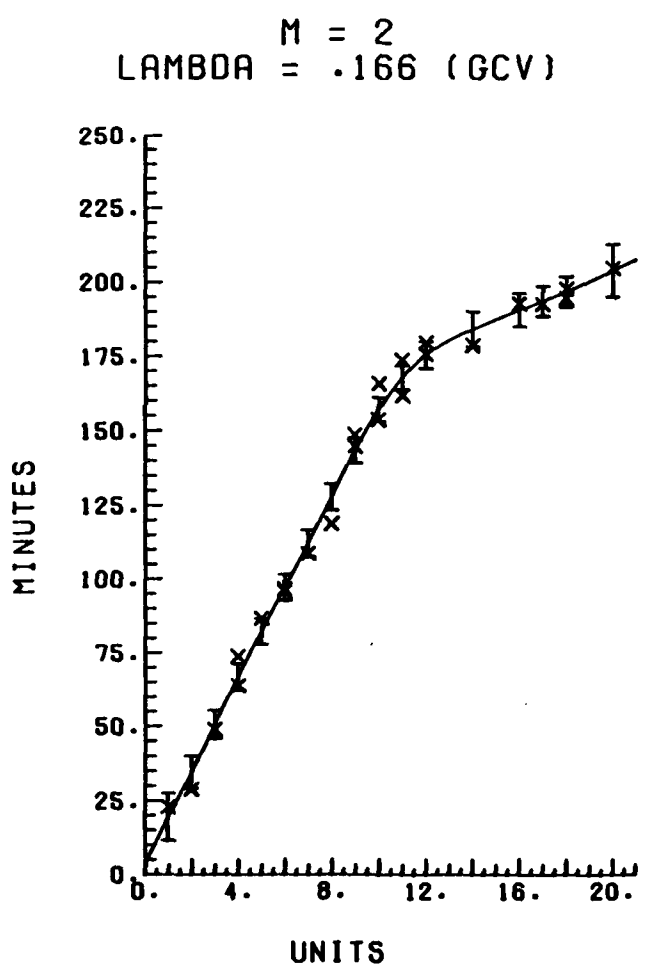


Figure 3.1c: Example 1 with $\lambda = .166$, the GCV choice

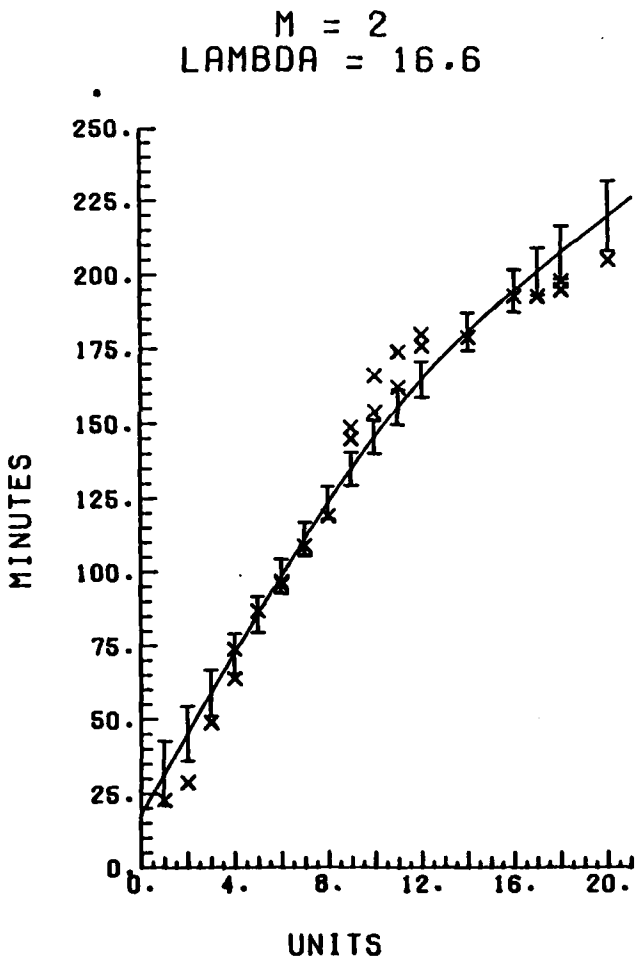


Figure 3.1d: Example 1 with $\lambda = 16.6$

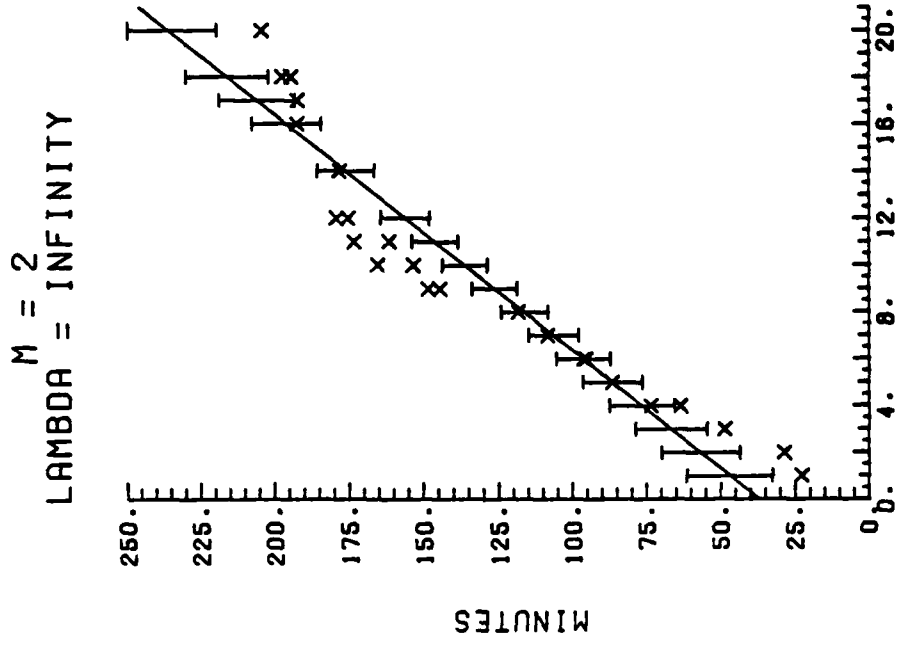


Figure 3.1e: Example 1 with $\lambda = \infty$

Considering the brief explanation of the problem given here the GCV choice of λ, as used in Figure 3.1c, seems reasonable to use in predicting the number of minutes spent. The GCV choice of λ appears to be the most visually pleasing and consistent with how we would expect the number of minutes spent on a service call to be related to the number of computer components repaired.

4. Generalized Cross Validation

In the example of Section 3 the smoothing parameter λ is unknown. To determine an estimate of this parameter Craven and Wahba (1979) and Wahba and Wold (1979) have suggested the use of generalized cross validation. A short synopsis of the development of this method is given to enhance the understanding of it.

The method of cross validation (presented here as related to LSS's) is developed in response to the question: How well may one expect LSS's to predict the true functional value $g(\underset{\sim}{t})$ at some point $\underset{\sim}{t}$?

Simple cross validation (SCV) suggests predicting the true functional values of data different from that used in the analysis to assess this predictive ability. In its simplest form this entails dividing the sample into two pieces of similar size using one section for optimization and the other for testing. In addition to this, in order to gain more information from the data, the two pieces may be interchanged and the optimization and testing performed on each.

SCV is alright if there is an ample supply of data so that halving or doubling it has little effect on the quality of the estimator. To lessen this effect Mosteller and Tukey (1968) propose single cross validation (1CV), (called ordinary cross validation by Wahba (1979)), which is described suitably by them as follows:

> "Suppose that we set aside one individual case, optimize for what is left, then test on the set-aside case. Repeating this for every case squeezes the data almost dry. If we have to go through the full

optimization calculation every time, the extra computation may be hard to face. Occasionally, one can easily calculate, either exactly or to an adequate approximation, what the effect of dropping a specific and very small part of the data will be on the optimized result. This adjusted optimized result can then be compared with the values for the omitted individual. That is, we make one optimization for all the data, followed by one repetition per case of a much simpler calculation, a calculation of the effect of dropping each individual, followed by one test of that individual. When practical, this approach is attractive."

To describe 1CV mathematically we require some notation. Let $g_\lambda(j)$ be the solution to the minimization of (2.1) with the j^{th} point removed from the analysis. Similarly, $D_\sigma(j)$ is the N-1 by N-1 matrix composed of D_σ with its j^{th} row and column removed. To "test on the set aside case" we require that $[(g_\lambda(j)(\underset{\sim}{t}_j) - z_j)/\sigma_j]^2$ be small. "Repeating this for every case" and averaging to yield an overall test gives

$$V_m^0(\lambda) = (1/N) \sum_{j=1}^{N} [(g_\lambda(j)(\underset{\sim}{t}_j) - z_j)/\sigma_j]^2 . \qquad (4.1)$$

1CV uses the λ which minimizes $V_m^0(\lambda)$.

To minimize $V_m^0(\lambda)$ directly is not a trivial computational matter. For each proposed value of λ a system of the form (2.8) and (2.9) (of order N+M-1 instead of N+M) must be solved for each of the N values left out of the analysis. This entails solving a linear system of order N+M-1 N times! As noted earlier "if we have to go through the full optimization calculation every time, the extra computation may be hard to face." Following the idea of Mosteller and Tukey we seek a computational simplification for the minimizer of $V_m^0(\lambda)$.

The simplified form for 1CV was first noted by Craven and Wahba (1979) and given in a slightly more general form in Wahba and Wendelberger (1980). The 1CV function may be written

$$V_m^0(\lambda) = (1/N) \sum_{j=1}^{N} [(g_\lambda(t_j) - z_j)/(\sigma_j(1-a_{jj}(\lambda)))]^2 . \qquad (4.2)$$

$a_{jj}(\lambda)$ is the j^{th} diagonal element of $A_m(\lambda)$ which is defined by

$$A_m(\lambda)z = \begin{pmatrix} g_\lambda(t_1) \\ \vdots \\ g_\lambda(t_N) \end{pmatrix}$$

where g_λ is the solution of (2.1). $A_m(\lambda)$ may be thought of as mapping the vector z into the smoothed values.

In this form "we make one optimization for all the data" by calculating g_λ then "followed by one repetition per case of a much simpler calculation, a calculation of the effect of dropping each individual." Here find $a_{jj}(\lambda)$ and use (4.2).

Evaluation of this formulation of $V_m^0(\lambda)$ involves solving a linear system of size N+M to find g_λ and one of size N to find $a_{jj}(\lambda)$. This is a considerable improvement over that of using (4.1) directly. Because of a mathematical simplification the amount of computation needed to minimize $V_m^0(\lambda)$ can be substantially reduced. From a practical point of view this makes the use of cross validation very attractive.

When applying cross validation to problems other than LSS's this last step of finding "what the effect of dropping a specific and very small part of the data will be on the optimized result" is very important and should not be

overlooked. In fact, this step often makes cross validation computationally feasible whereas without this insight it may be impractical.

Finding the minimizer of $V_m^0(\lambda)$ requires its evaluation at different values of λ as determined by a search routine. Hence, although the minimization is possible we need to repeatedly solve large linear systems with the number of solution times being a function of the search routine employed.

In $V_m^0(\lambda)$ of (4.1) each deviation of $g_\lambda^{(i)}(\underset{\sim}{t}_i)$ from the observed value z_i is treated symmetrically. This choice is arbitrary and is chosen for simplicity. A more general approach is to weight each term of (4.1) or equivalently (4.2) to yield

$$V_m(\lambda) = (1/N) \sum_{i=1}^{N} W_i[(g_\lambda(\underset{\sim}{t}_i) - z_i)/(\sigma_i(1-a_{ii}(\lambda)))]^2. \qquad (4.3)$$

Before a discussion of the choice of these weights the following definition is needed.

Definition:

$$R_m(\lambda) = E(1/N) \sum_{i=1}^{N} [(f(\underset{\sim}{t}_i) - g_\lambda(\underset{\sim}{t}_i))/\sigma_i]^2$$

is the expected weighted (by σ_i) mean squared error between the true function (f) and the spline (g_λ) evaluated at the independent variables ($\underset{\sim}{t}_i$). Here E denotes mathematical expectation with respect to the error distribution of the random errors as described in the model of Section 2.

If we want $R_m(\lambda)$ to be small then the generalized cross validation value of λ should be used as the smoothing parameter value. Using 1CV as motivation

Craven and Wahba (1979) and Golub, Heath and Wahba (1979) have shown that the λ which minimizes $V_m(\lambda)$ with weights

$$W_i = (1-a_{ii}(\lambda))^2/(1-N^{-1}\sum_{j=1}^{N} a_{jj}(\lambda))^2$$

is an estimate of the λ which minimizes $R_m(\lambda)$. Using these weights in (4.3) gives the generalized cross validation function (GCVF)

$$V_m(\lambda) = (1/N)\sum_{i=1}^{N}[(g_\lambda(\underset{\sim}{t}_i)-z_i)/(\sigma_i(1-N^{-1}\sum_{j=1}^{N} a_{jj}(\lambda)))]^2 \ . \tag{4.4}$$

The minimizer of (4.4) is called the GCV estimate of λ.

The GCVF can be rewritten as

$$V_m(\lambda) = (1/N)||D_\sigma^{-1}(I - A_m(\lambda))\underset{\sim}{z}||^2/((1/N)Tr(I-A_m(\lambda)))^2 \ ; \tag{4.5}$$

where Tr is the trace.

Wahba (1981) has proposed

$$\sigma_e^2 = ||D_\sigma^{-1}(I-A_m(\lambda))\underset{\sim}{z}||^2/Tr(I-A_m(\lambda)) \tag{4.6}$$

as an estimate of the error variance σ^2. This leads us to consider $df_e = Tr(I-A_m(\lambda))$ as the degrees of freedom of error. Using these notions we rewrite the GCVF as

$$V_m(\lambda) = N\sigma_e^2/df_e \ . \tag{4.7}$$

The method of GCV may be viewed as minimizing the estimated error variance per error degrees of freedom. This may further be thought of as a form of parsimonious model selection.

In the next section we see that the computation of $V_m(\lambda)$ is reduced to essentially the singular value (or eigenvalue-eigenvector) decomposition of a symmetric positive definite N-M by N-M matrix (M is usually a small integer). The above decomposition makes it possible to form $V_m(\lambda)$ by simple scalar operations for each value of λ. Thus we have taken the ideas of Mosteller and Tukey one step further. This algorithm is much simpler than the original analysis at essentially the cost of a one time eigenvalue-eigenvector decomposition; i.e., changing the dependent variable (but not the independent variables) does not necessitate another spectral decomposition. Thus, many data sets which have identical independent variables but different dependent variables may be analyzed quite easily and inexpensively.

When using GCV with a small sample size we may run into problems. The most frequent small sample problem with GCV is that $\lambda = 0$ or $\lambda = \infty$ is chosen when physical considerations dictate that it should not be. $\lambda = 0$ implies that we are interpolating the dependent variable. This should be done if the true underlying rigidity ρ is zero. λ equal to infinity implies that we are fitting a polynomial of degree m-1 by least squares. This should be done if either the variance is large (relative to the dependent variable) or if the true underlying rigidity is infinite (i.e., the true model is a polynomial). If it is clear from other considerations that the value of λ chosen is not indicative of the actual underlying mechanism then that particular value should not be used and the model assumptions should be checked for violations.

The choice of m can also be made by GCV, see Lucas (1978) and Wahba and Wendelberger (1980).

5. Algorithm

The user must supply N independent variables, $\underset{\sim}{t}_i \epsilon R^d$, i=1,...,N, and their corresponding dependent variables, $z_i \epsilon R^d$, i=1,...,N to compute the LSS at a point $\underset{\sim}{t} \epsilon R^d$. Assume that the model described in Section 2 holds. In particular, assume the independent variables $\underset{\sim}{t}_i$ are known without error and the dependent variables z_i consist of the true function value at $\underset{\sim}{t}_i$, $f(\underset{\sim}{t}_i)$, plus "noise," e_i, $z_i = f(\underset{\sim}{t}_i) + e_i$. The e_i are independent with finite variance $\sigma^2\sigma_i^2$, σ^2 an unknown constant.

To produce the coefficients $\underset{\sim}{c}$ and $\underset{\sim}{d}$ needed to evaluate the spline we solve the linear system of equations

$$(K + N\lambda^* D_\sigma^2)\underset{\sim}{c} + T\underset{\sim}{d} = \underset{\sim}{z}$$

and

$$T^T\underset{\sim}{c} = 0 .$$

In this system λ^* is the optimal value of the smoothing parameter λ as determined by the generalized cross validation function. If λ^* is known then the solution of the above linear system could be accomplished for relatively large values of N. However, it is usually unknown and must be calculated in order to solve the system of equations.

The method currently used to determine λ^* requires the solution of a symmetric N-M dimensional eigenvalue-eigenvector problem. This is the current computational barrier to solving problems with large numbers of observations.

The algorithm presented in Wahba and Wendelberger (1980) requires the inversion of a matrix of order M and two eigenvalue-eigenvector decompositions of symmetric matrices, one N by N and the other (positive definite)

N-M by N-M. The algorithm presented here requires the solution of a triangular system of order M, the QR-decomposition of an N by M matrix and the singular value (or eigenvalue-eigenvector) decomposition of a symmetric positive definite N-M by N-M matrix. This algorithm is faster and requires fewer operations, primarily because of the replacement of one N by N eigenvalue-eigenvector decomposition by the QR-decomposition of an N by M matrix ($M < N$).

This algorithm provides for replicated points. A replicated point is one for which there is more than one observation of the dependent variable for a particular value of the independent variable. Let the total number of unique (independent variable) points be N_N and define $N_O = N - M - N_N$. Then the computational algorithm is as follows:

(i) Compute $T_\sigma = D_\sigma^{-1}T$.

(ii) Perform the QR-decomposition described in Dongarra, et al., (1979), of $T\sigma$.

$$T_\sigma = (Q_1,Q_2) \times (R^T,0)^T .$$

(iii) Calculate $B = Q_2^T D_\sigma^{-1} K D_\sigma^{-1} Q_2$.

(iv) Decompose $B = (U_1,U_2)D_B'(U_1,U_2)^T$,

using the singular value decomposition of B, as described by Golub and Reinsch (1970) or using the spectral decomposition of B as described by Smith, et al., (1976); where

$D_{B'}$ - diagonal matrix of the eigenvalues (b_i) of B, which is of dimension N-M by N-M,

D_B - diagonal matrix of the positive eigenvalues (b_i) of B, which is of dimension N_N by N_N,

U_1 - the eigenvectors of the positive eigenvalues of B, which is of dimension (N-M) by N_N, and

U_2 - the eigenvectors of the zero eigenvalues of B, which is of dimension (N-M) by N_O.

(v) Form $\underset{\sim}{w} = U_1^T Q_2^T D_\sigma^{-1} \underset{\sim}{z}$,

$\underset{\sim}{w}^T = (w_1, \ldots, w_{N_N})$.

(vi) Obtain λ^* as the minimizer of

$$V(\lambda) = \begin{cases} N \sum_{i=1}^{N_N} [w_i/(b_i/N + \lambda)]^2 / \left(\sum_{i=1}^{N_N} (1/(b_i/N+\lambda)) \right)^2, \\ \quad \lambda \neq \infty \text{ and } N-M = N_N \\ N[\underset{\sim}{z}_\sigma^T Q_2 Q_2^T \underset{\sim}{z}_\sigma - \underset{\sim}{w}^T \underset{\sim}{w} + \lambda^2 \sum_{i=1}^{N_N} (w_i/(b_i/N+\lambda))^2] + \\ \quad (N-M-N_N+\lambda \sum_{i=1}^{N_N} (1/(b_i/N+\lambda)))^2, \\ \quad \lambda \neq \infty \text{ and } N-M \neq N_N \\ N \underset{\sim}{z}_\sigma^T Q_2 Q_2^T \underset{\sim}{z}_\sigma/(N-M)^2, \quad \lambda = \infty \end{cases} \tag{5.1}$$

where $\underset{\sim}{z}_\sigma = D_\sigma^{-1} \underset{\sim}{z}$.

(vii) Calculate

$$\underset{\sim}{c} = \begin{cases} D_\sigma^{-1}Q_2U_1D_B^{-1}U_1^TQ_2^T\underset{\sim}{z}_\sigma , & \lambda = 0 \\ D_\sigma^{-1}Q_2U_1[(D_B+N\lambda I)^{-1}]U_1^TQ_2^T\underset{\sim}{z}_\sigma , & \\ \quad 0 < \lambda < \infty \text{ and } N-M = N_N & \\ D_\sigma^{-1}Q_2U_1[(D_B+N\lambda I)^{-1}-(N\lambda)^{-1}I]U_1^TQ_2^T\underset{\sim}{z}_\sigma & \\ \quad +(N\lambda)^{-1}D_\sigma^{-1}Q_2Q_2^T\underset{\sim}{z}_\sigma , \quad 0 < \lambda < \infty \text{ and } N-M \neq N_N . & \\ \underset{\sim}{0} , \qquad \lambda = \infty . & \end{cases} \tag{5.2}$$

(viii) Solve the triangular system.

$$R\underset{\sim}{d} = Q_1^TD_\sigma^{-1}(\underset{\sim}{z} - K\underset{\sim}{c}) \text{ for } \underset{\sim}{d},$$

$$\underset{\sim}{d}^T = (d_1,\ldots,d_M) .$$

An important aspect of this method is the relatively small cost of reconstructing a new LSS using the identical independent variables while changing only the dependent variables. To see this notice that the bulk of the computational effort is in steps (i) through (iv) which do not require knowledge of the dependent variables. These steps depend upon the independent variables and D_σ. To construct a second LSS with the same independent variables and identical D_σ we need only save the matrices U_1, D_σ, D_B, Q_1, Q_2 and R. With these matrices we perform steps (v) through (viii) to produce a spline for another set of dependent variables, say $\underset{\sim}{z}'$, with little additional computational effort.

The fact that obtaining another spline from $\underset{\sim}{z}'$ is easy requires further consideration. It is made possible because of the necessity to minimize the

GCVF. This minimization provides the mechanism to easily calculate $\underset{\sim}{c}$ and $\underset{\sim}{d}$ in steps (vii) and (viii) of the algorithm. If λ^* was somehow known a priori then we could go right ahead and solve the linear system (2.8) and (2.9) at a much less one time cost. However, even with λ^* known, if we had many new data sets $\underset{\sim}{z}'$ then for some number of them it indeed would be easier to do the spectral decomposition once and for all.

Instead of saving U_1, D_σ, D_B, Q_1, Q_2 and R we actually save Q_2U_1, D_σ, D_B, $Q_1^TD_\sigma^{-1}K$ and the QR-decomposition of T_σ to retrieve R, $Q_2Q_2^T$ and Q_1. By using these matrices we can perform steps (v) through (viii) quite inexpensively. The QR-decomposition can be stored in the storage which has been allocated for T_σ plus M additional storage locations. $Q_1^TD_\sigma^{-1}K$ is retained so that it is unnecessary to reevaluate K.

6. Example 2--Franke's Principal Test Function, d=2.

Example 2 is a Monte Carlo experiment to demonstrate the surface (d=2) which may be obtained by using an LSS with GCV. The "principal test function" of Franke (1979) is used as the true function f. This surface consists of two Gaussian peaks and one Gaussian dip superimposed on a surface sloping towards the first quadrant. The surface is defined by

$$f(x,y) = .75 \exp -[[(9x-2)^2+(9y-2)^2]/4]$$
$$+ .75 \exp -[[(9x+1)^2/ 49]+[(9y+1)/10]]$$
$$+ .50 \exp -[[(9x-7)^2+(9y-3)^2]/4]$$
$$- .20 \exp -[(9x-4)^2+(9y-7)^2]$$

A plot of the surface f is given in Figure 6.1.

The surface is reconstructed from 169 "noisy" observations on the grid

$$G = \{\underset{\sim}{t}_i | \underset{\sim}{t}_i = (\frac{2j-1}{26}, \frac{2k-1}{26}),\ i=13(j-1)+k;\ j,k=1,\ldots,13\} .$$

The "noisy" observations are

$$z_i = f(t_i) + e_i \text{ with } e_i \sim N(0,\sigma^2),\ i=1,\ldots,169,\ \sigma^2=(.03)^2.$$

The e_i are generated by the pseudo random number generator RAENBR at the Madison Academic Computing Center, MACC (1978). The LSS with m=2 and the smoothing parameter chosen by GCV is plotted in Figure 6.2. The closeness of fit can be qualitatively seen by overlaying Figure 6.2 on Figure 6.1.

For this example the calculated $\sigma_e^2=(.026)^2$, (using (4.6)), compares favorably with the true $\sigma^2=(.03)^2$. Using σ_e^2 to obtain confidence intervals for the true curve at the grid points G as in Wahba (1981) gives the 95% confidence intervals

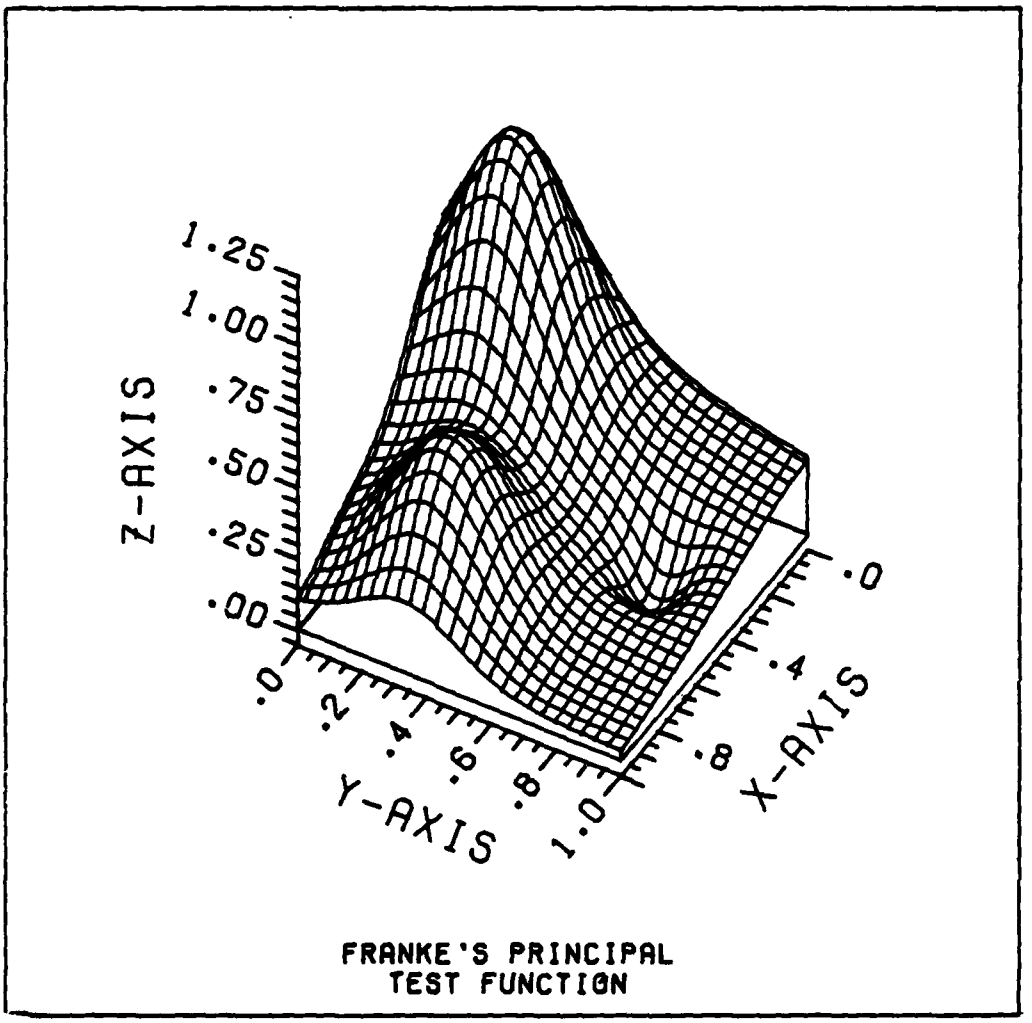


Figure 6.1: Example 2--Franke's Principle Test Function

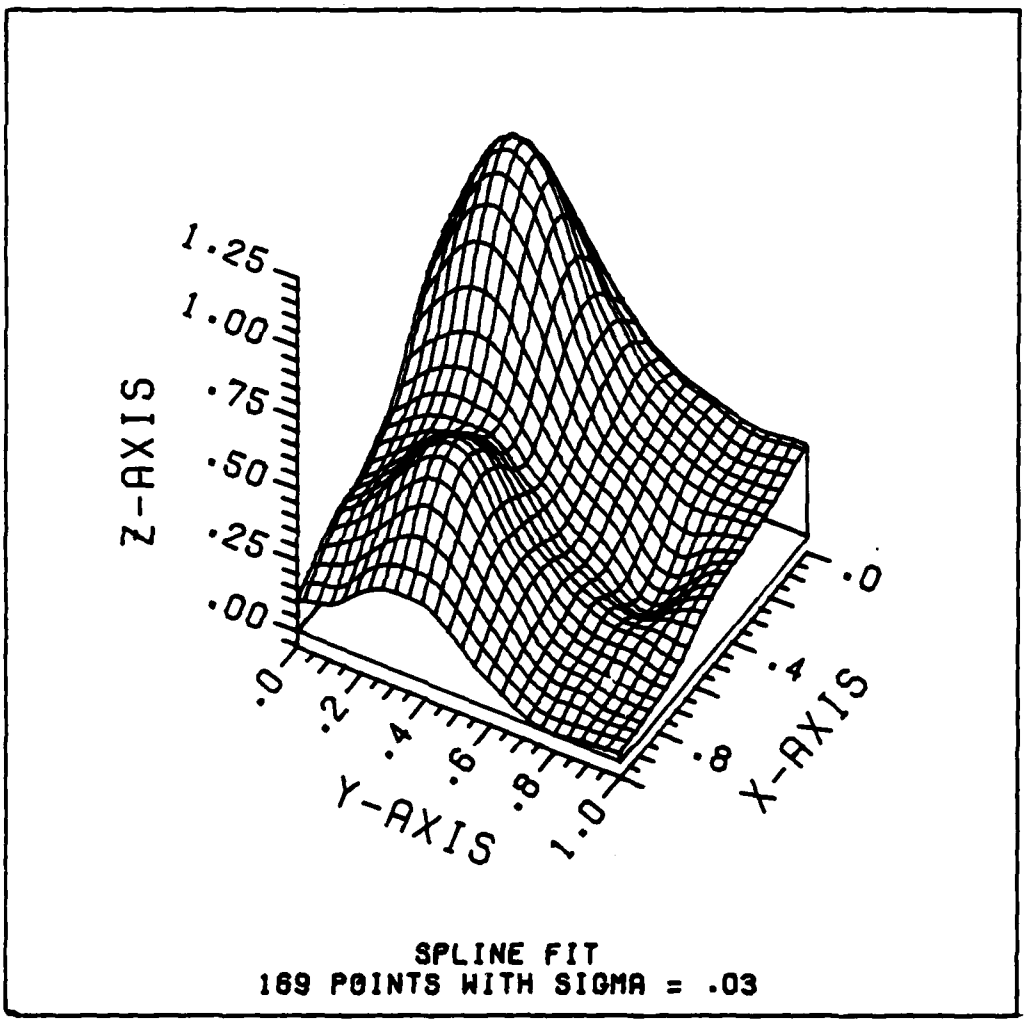


Figure 6.2: Plot of the m = 2, GCV λ, spline fit to Franke's Principal Test Function from 169 "noisy" points with σ = .03

$$g_\lambda^*(t_i) \pm 1.96\sigma_e\sigma_i(a_{ii}(\lambda^*))^{1/2}\ , \quad i=1,\ldots,N.$$

Figure 6.3 gives the cross section along the grid showing the true curve, spline fit, observation and 95% confidence interval at each point for each value of x_j, $i=1,\ldots,13$.

The number of 95% confidence intervals which cover the true surface is known because the true surface is known. For this example 162 or 95.9% of the intervals cover the true surface. This is a favorable comparison since the expected number is 161. This example was not chosen because of this agreement but rather was the only one run by prior decision.

The example given here uses points on a grid only for clarity of display. For other d=2 Monte Carlo results see Wahba and Wendelberger (1980). The meteorological example given there uses irregularly spaced points.

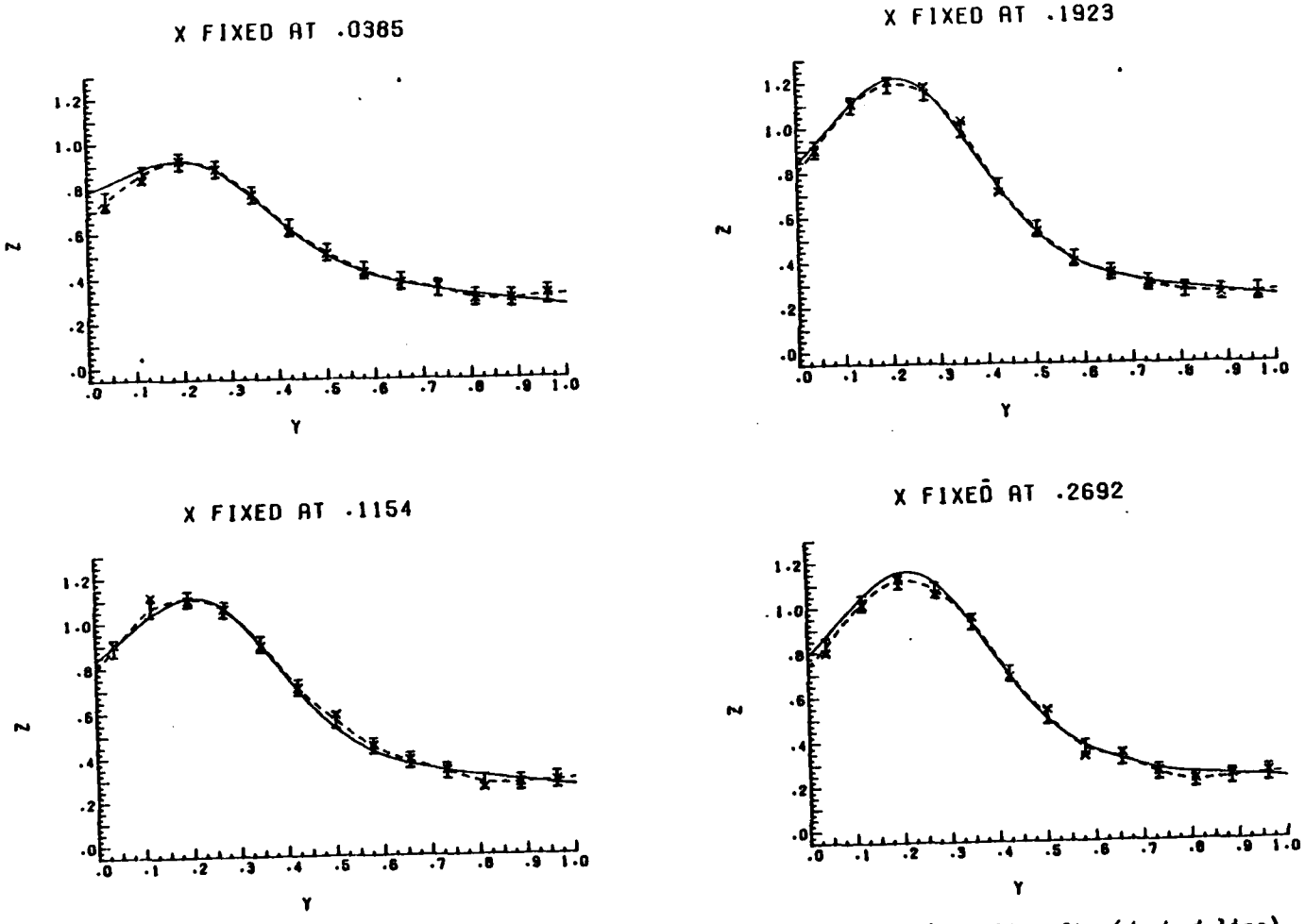


Figure 6.3: 13 Cross sections of Example 2--true curve (solid line), spline fit (dashed line)

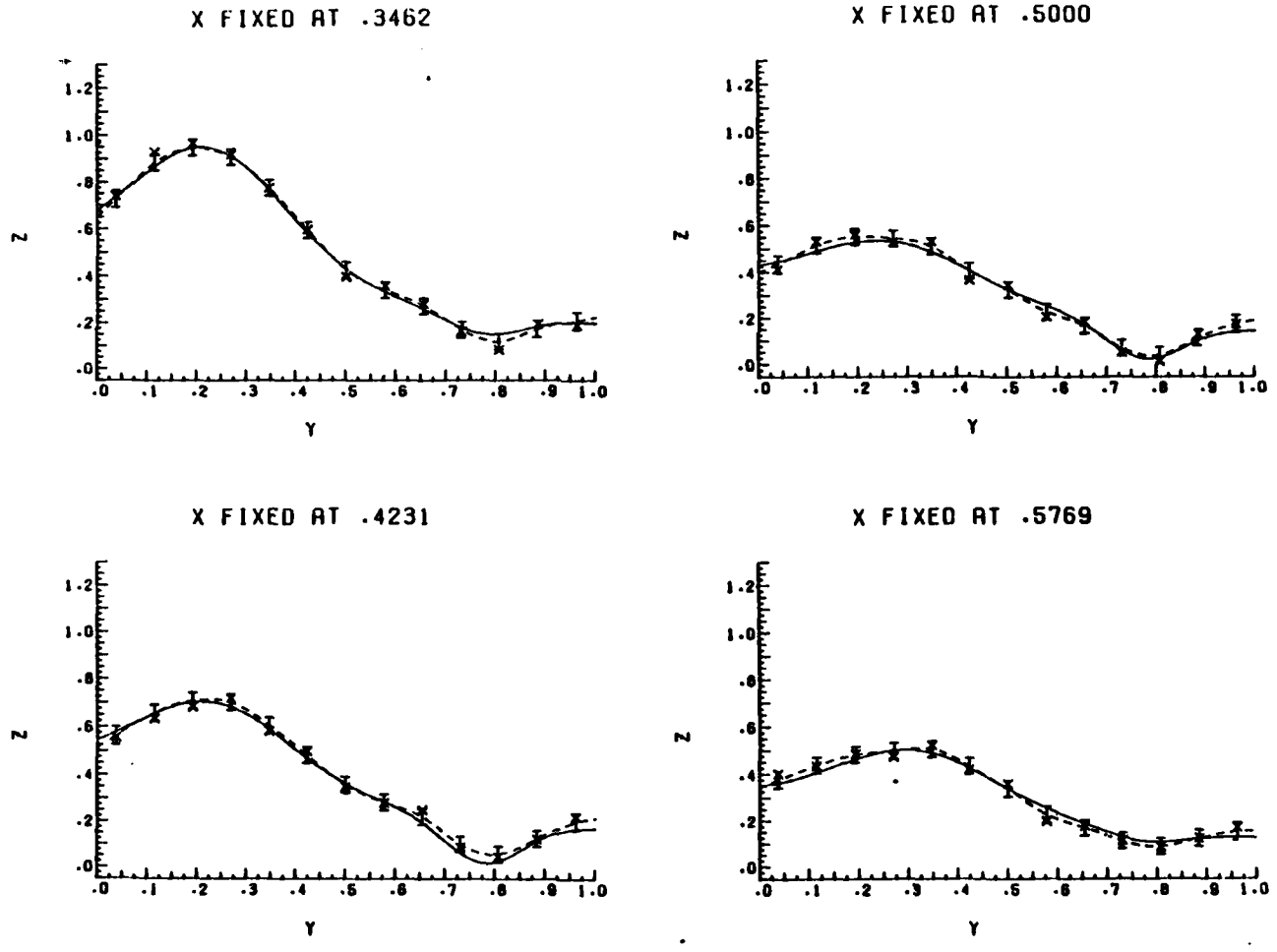


Figure 6.3: 13 Cross sections of Example 2--true curve (solid line), spline fit (dashed line)

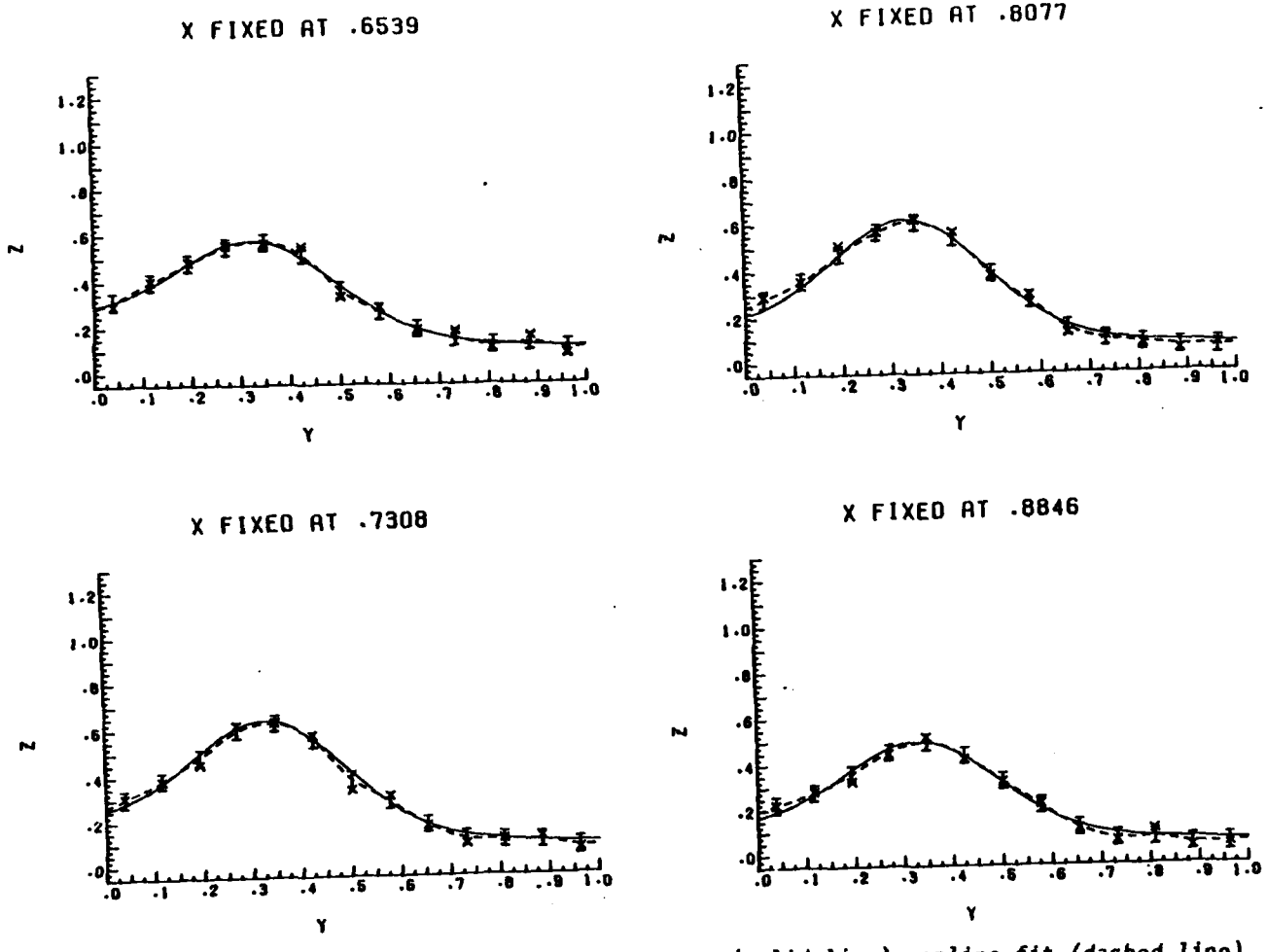


Figure 6.3: 13 Cross sections of Example 2--true curve (solid line), spline fit (dashed line)

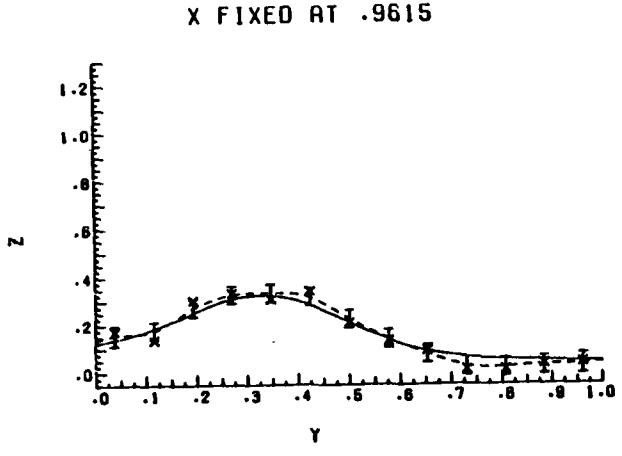


Figure 6.3: 13 Cross sections of Example 2--true curve (solid line), spline fit (dashed line)

7. Example 3—Derivatives and Outliers, d=3.

Example 3 is a Monte Carlo experiment with d=3 and true function

$f(x_1,x_2,x_3)=(2\pi)^{-3/2}\exp[(x_1^2+4x_2^2+9x_3^2)/(-2)]$.

Contours of f, f' and f'' are given as the solid lines in Figures 7.4, 7.5 and 7.6.

Three hundred points t_i, i=1,...,300 are taken from a uniform distribution in $R=\{(x_1,x_2,x_3)\,|\,-2<x_1<2,\ -1<x_2<1,\ -2/3<x_3<2/3\}$. The true function f is evaluated at each of the points t_i and added to a Gaussian pseudo random variable with standard deviation $\sigma=.0025$ to yield observation z_i. The peak height of f is approximately .0634. σ is roughly 4% of the peak height and therefore these data have a "typical" noise level.

A value of m=4 was chosen for this example in order that the second derivative of the spline could be used as an estimate of the second derivative of f. If k is the order of the derivative desired then 2m-2k-d must be positive. Here 2x4-2x2-3 = 1 > 0 and so the second derivative of the LSS will be a good estimate of the second derivative of f; for details see Wahba and Wendelberger (1980).

The estimate σ_e for this experiment is .0024 which agrees nicely with the true value of .0025.

Contours of the true function and the fitted spline, $g_{\lambda *}$, are plotted in Figure 7.4 for 4 values of x_3. Because of the symmetry of the true surface it was not plotted for negative values of x_3. The true function and the fitted spline are close to one another near the center of the region and this closeness degrades as we approach the boundary in each of the three directions.

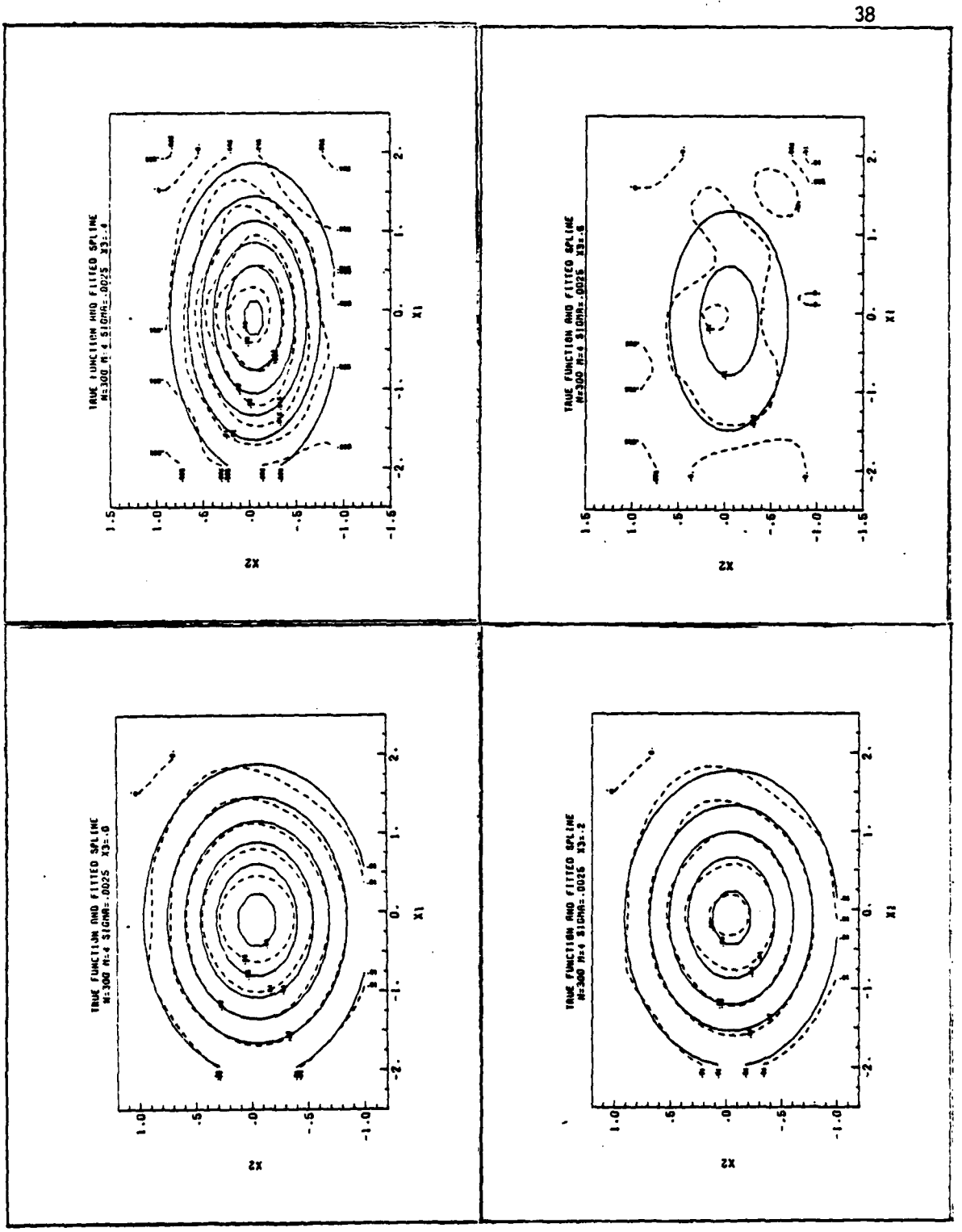


Figure 7.4: Example 3--solid line is f, dashed line is g.

The contours of the derivatives of f and g_λ^* with respect to x_1, x_2 and x_3 are given in Figures 7.5a, 7.5b and 7.5c, respectively. The contours of the second derivatives of f and g_λ^* with respect to x_1x_1, x_1x_2, x_1x_3, x_2x_2, x_2x_3 and x_3x_3 are given in Figures 7.6a, 7.6b, 7.6c, 7.6d, 7.6e, and 7.6f, respectively. The same qualitiative behavior is displayed by these derivatives as of the function with the degradation occurring relatively more rapidly as the boundary is approached. Figure 7.6f which is $(\partial^2)/(\partial x_3 \partial x_3)$ of f and g_λ^* displays a particularly good fit near the center of R.

LSS's may be utilized to detect outliers in multidimensional noisy data provided that the model of Section 2 is (nearly) appropriate. The model requires that the observations are unbiased, i.e., that $E\underset{\sim}{z}=\underset{\sim}{f}$. The errors should be additive and have a known relative error structure, D_σ. For the purpose of the outlier study here we shall further assume that each error e_i has a Gaussian distribution.

To what extent the assumption of normality may be relaxed in practice requires further study. The smoothness assumption requires that $f(\underset{\sim}{t})$ is a smooth function of $\underset{\sim}{t}$. This rules out "cliff" functions or those with discontinuities. By using a probability plot of the residuals the example discussed here, which satisfies the above requirements, will be used to demonstrate an outlier detection method.

Data sets with outliers need to be constructed. To accomplish this choose the two points of $\underset{\sim}{t}_i$, i=1,...,300 which are nearest to and farthest from the origin, which is the center of the data region. These two points are $\underset{\sim}{t}_k$ = (-.056, -.032, -.042) and t_l = (1.985, -.879, -.325), respectively. To

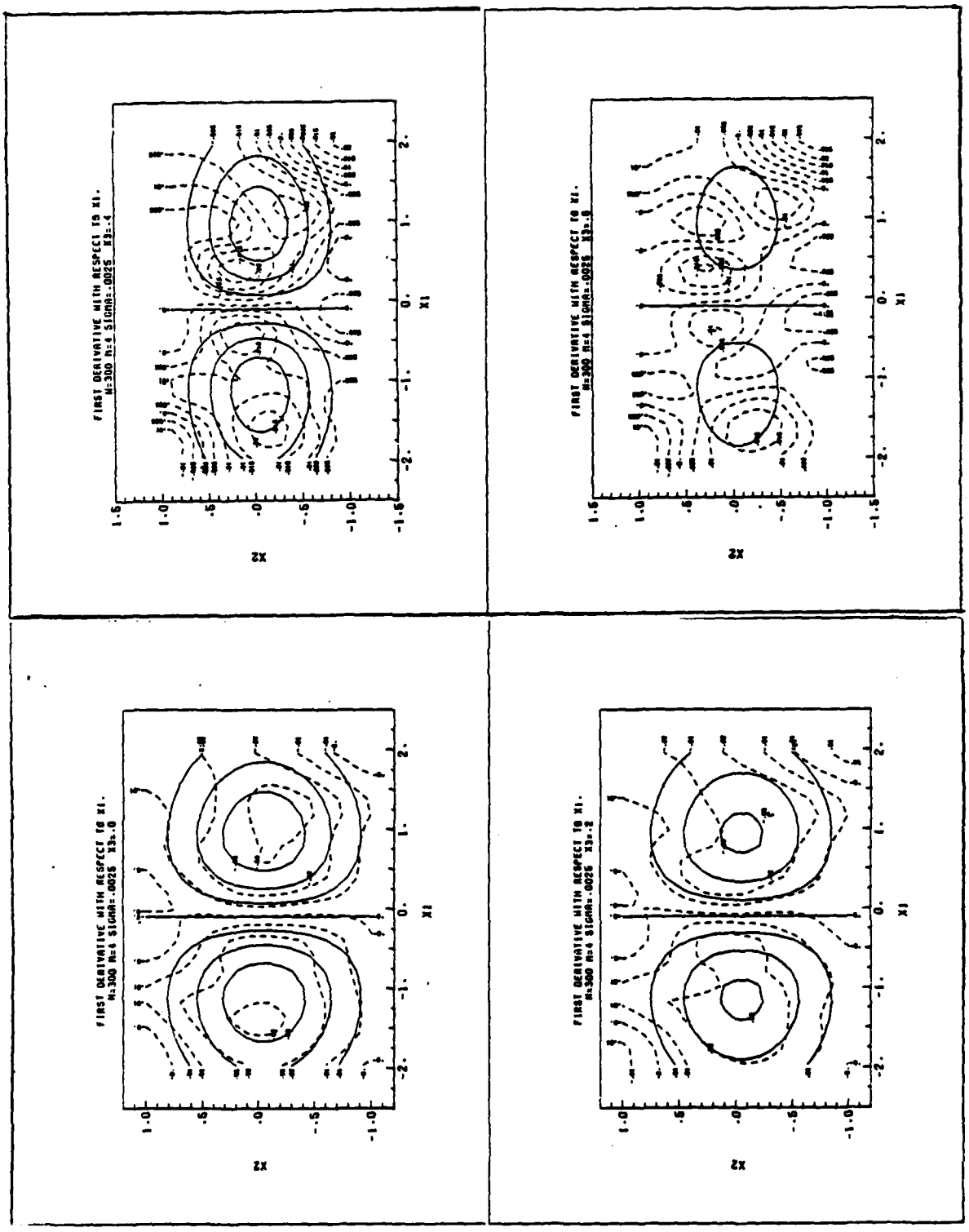


Figure 7.5a: Example 3--Solid line is df/dx_1, dashed line is dg/dx_1.

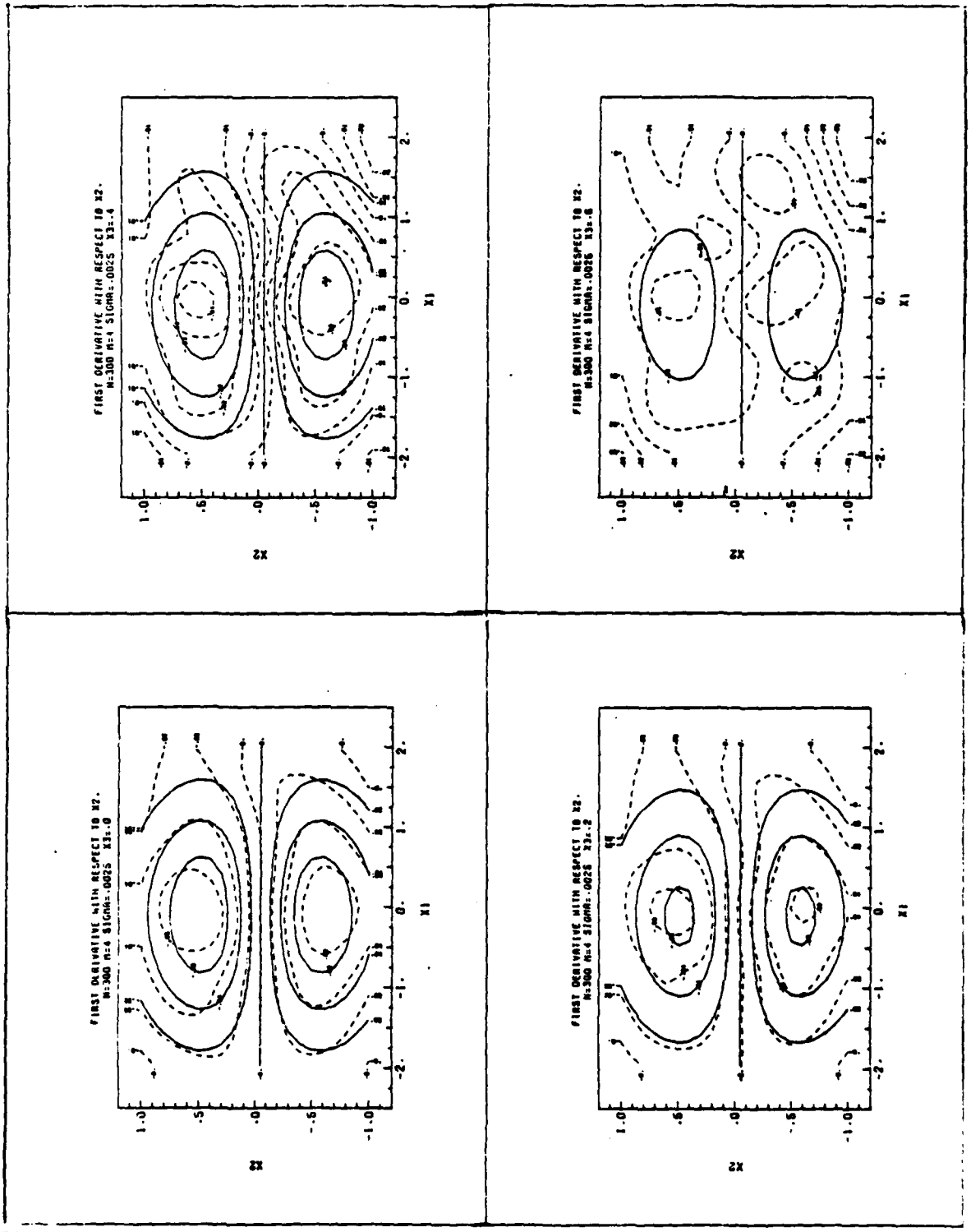


Figure 7.5b: Example 3--solid line is df/dx_2, dashed line is dg/dx_2.

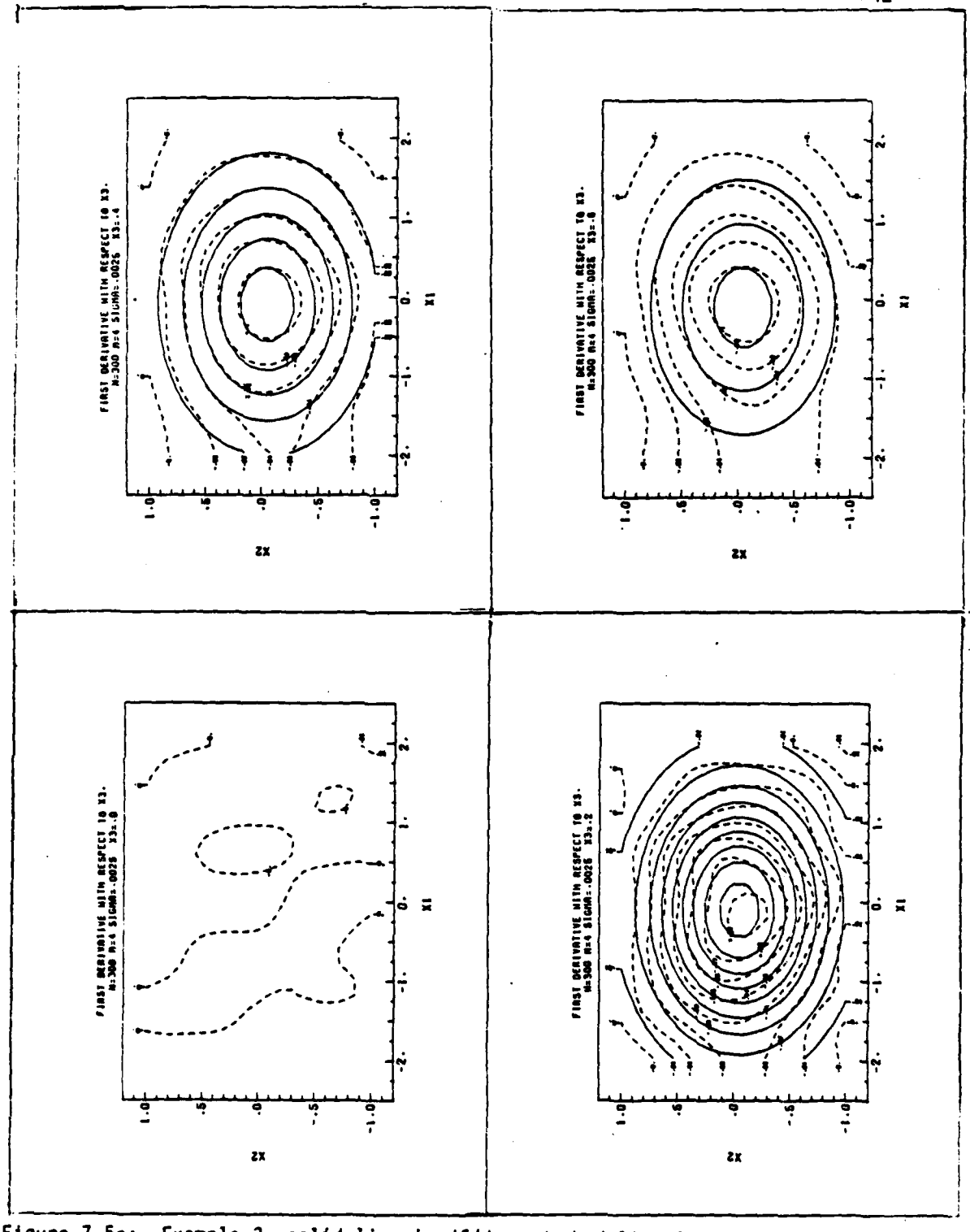


Figure 7.5c: Example 3--solid line is df/dx_3, dashed line is dg/dx_3.

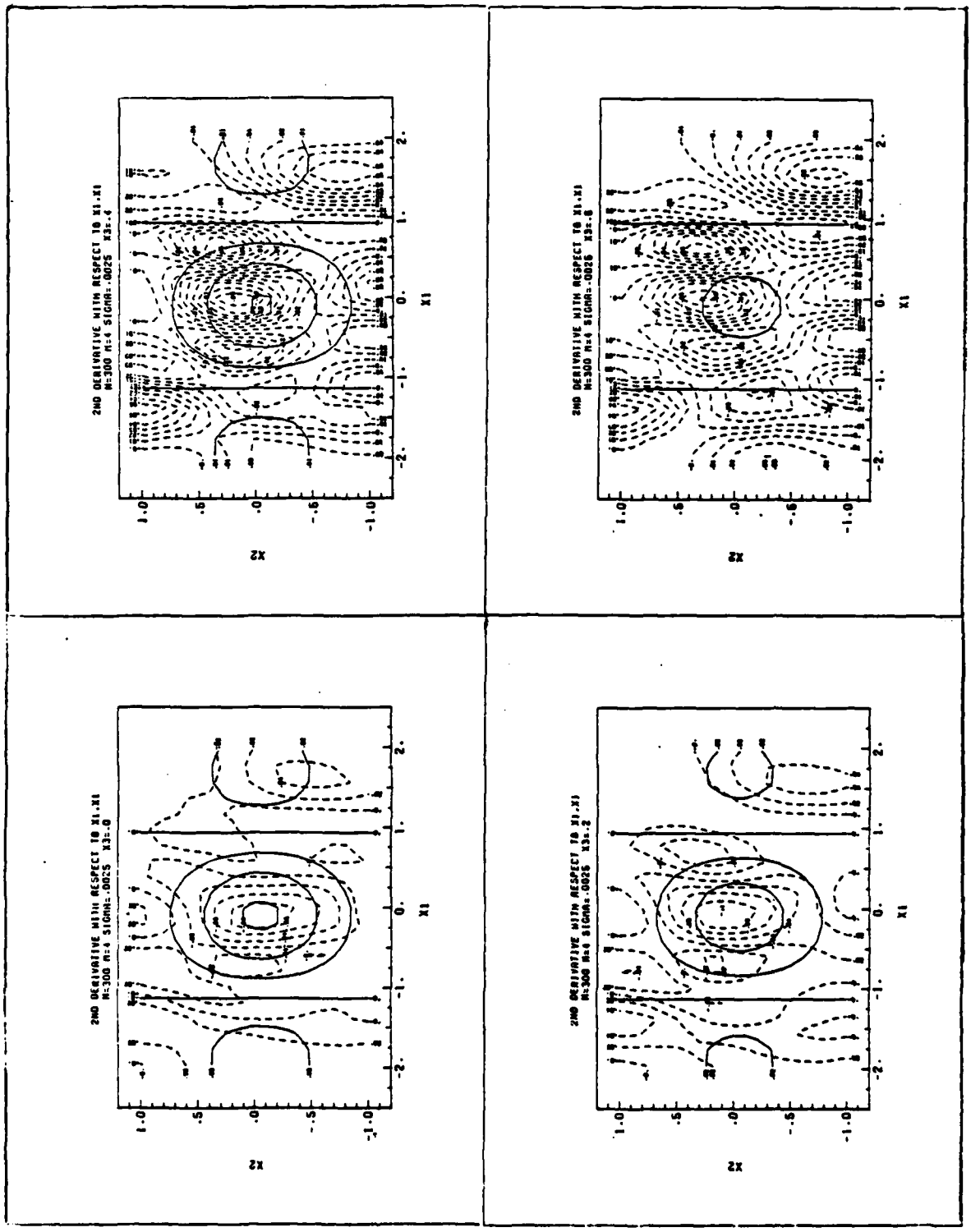


Figure 7.6a: Example 3--solid line is d^2f/dx_1^2, dashed line is d^2g/dx_1^2.

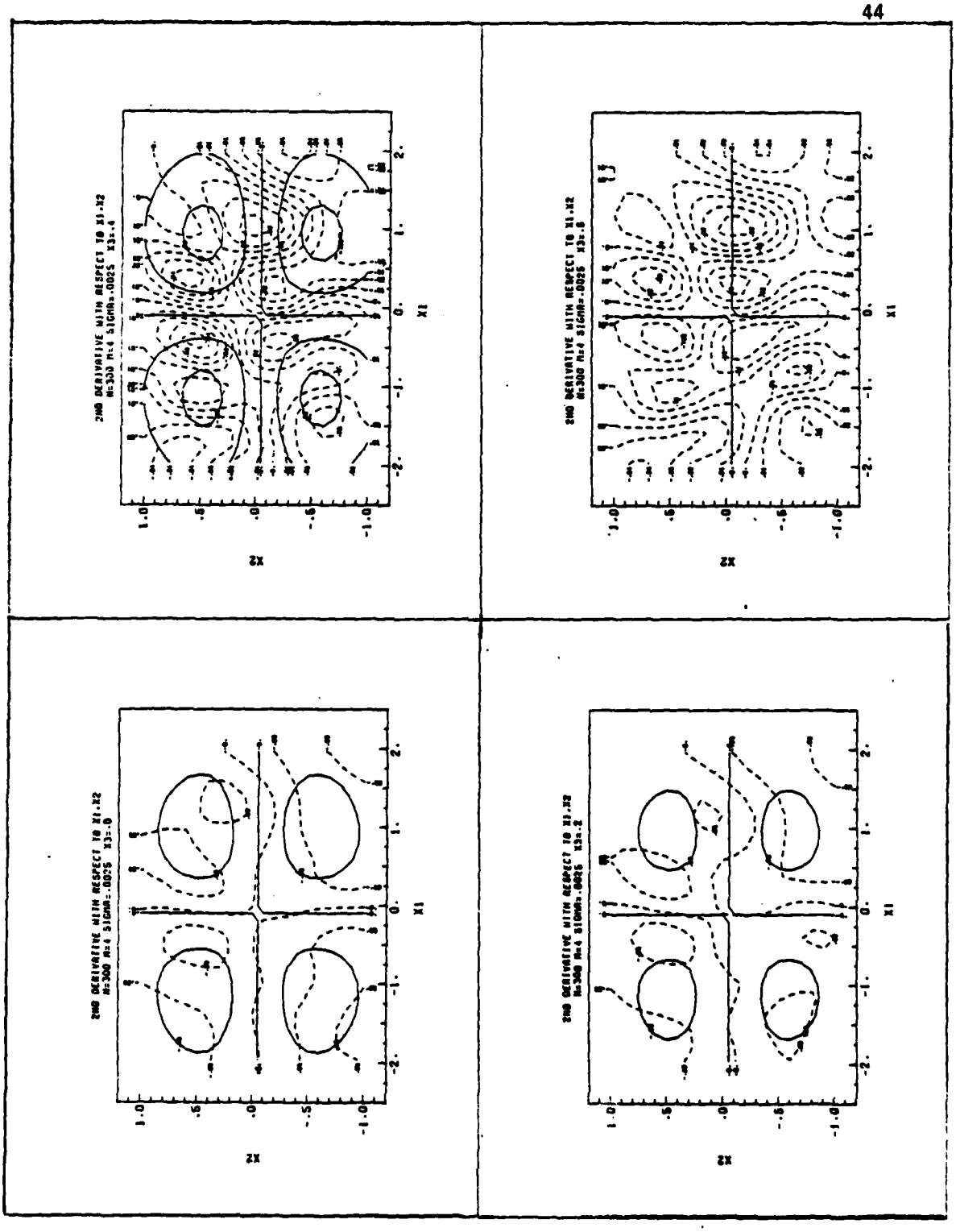


Figure 7.6b: Example 3--solid line is d^2f/dx_1dx_2, dashed line is d^2g/dx_1dx_2.

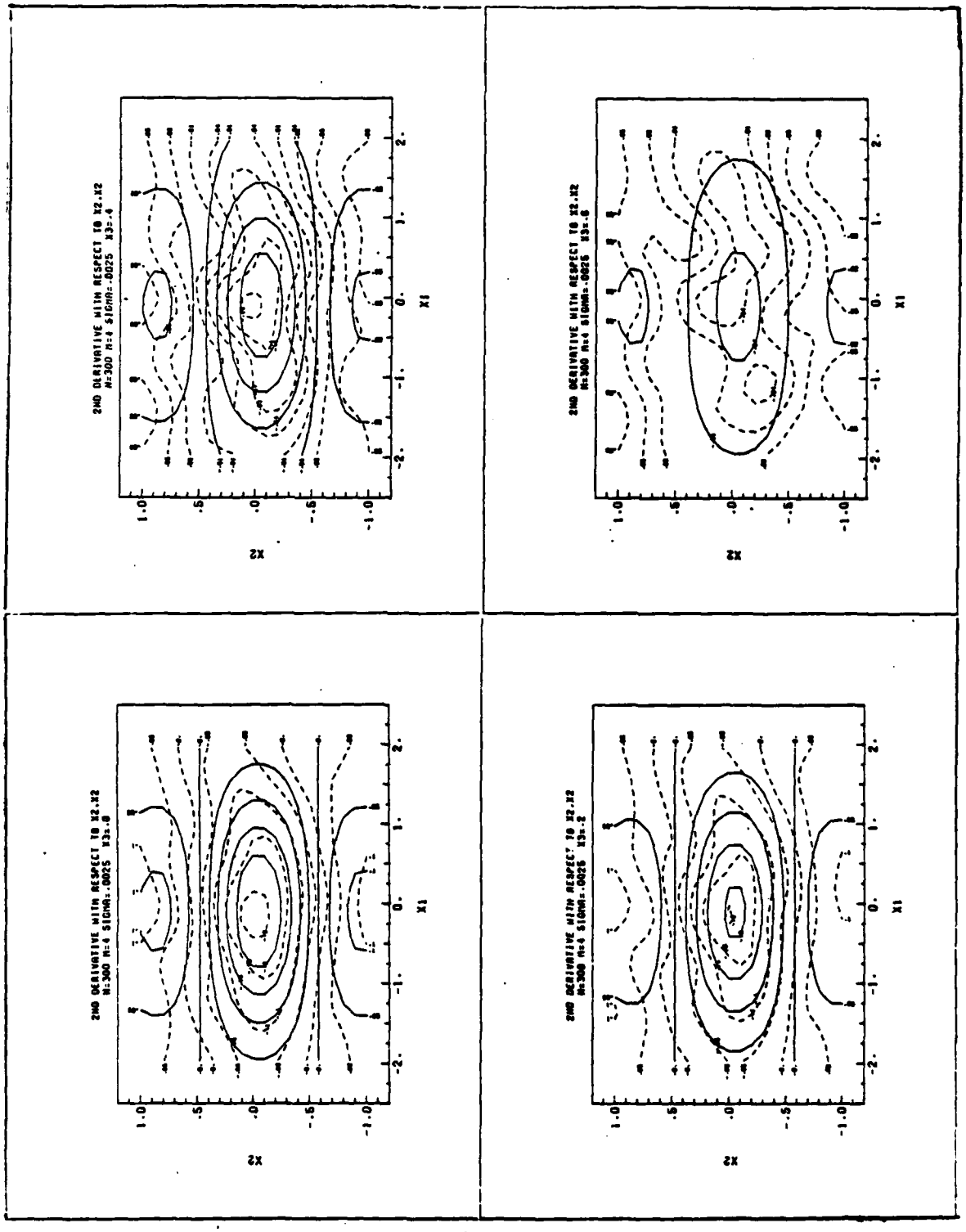


Figure 7.6c: Example 3--solid line is d^2f/dx_2^2, dashed line is d^2g/dx_2^2.

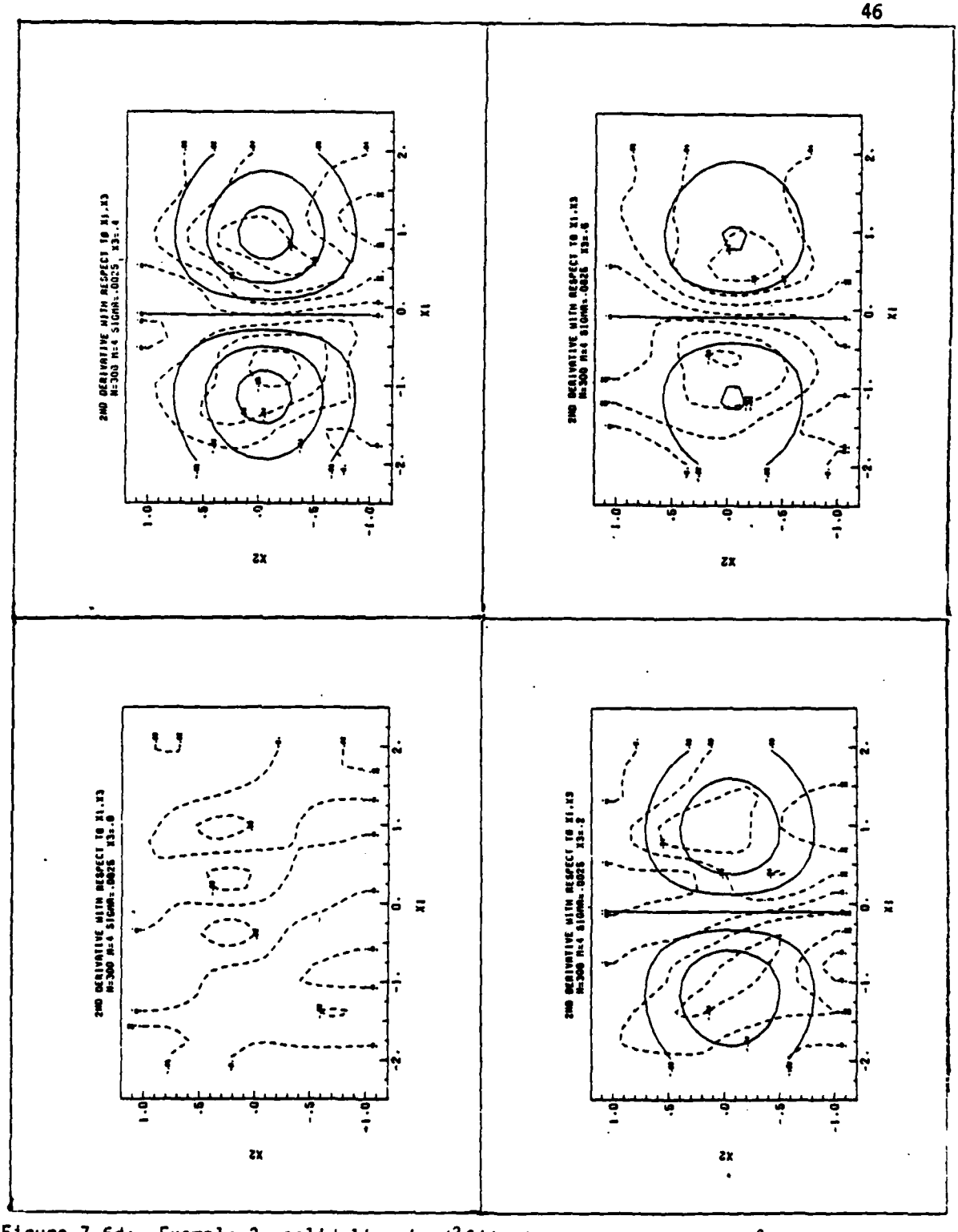


Figure 7.6d: Example 3--solid line is d^2f/dx_1dx_3, dashed line is d^2g/dx_1dx_3.

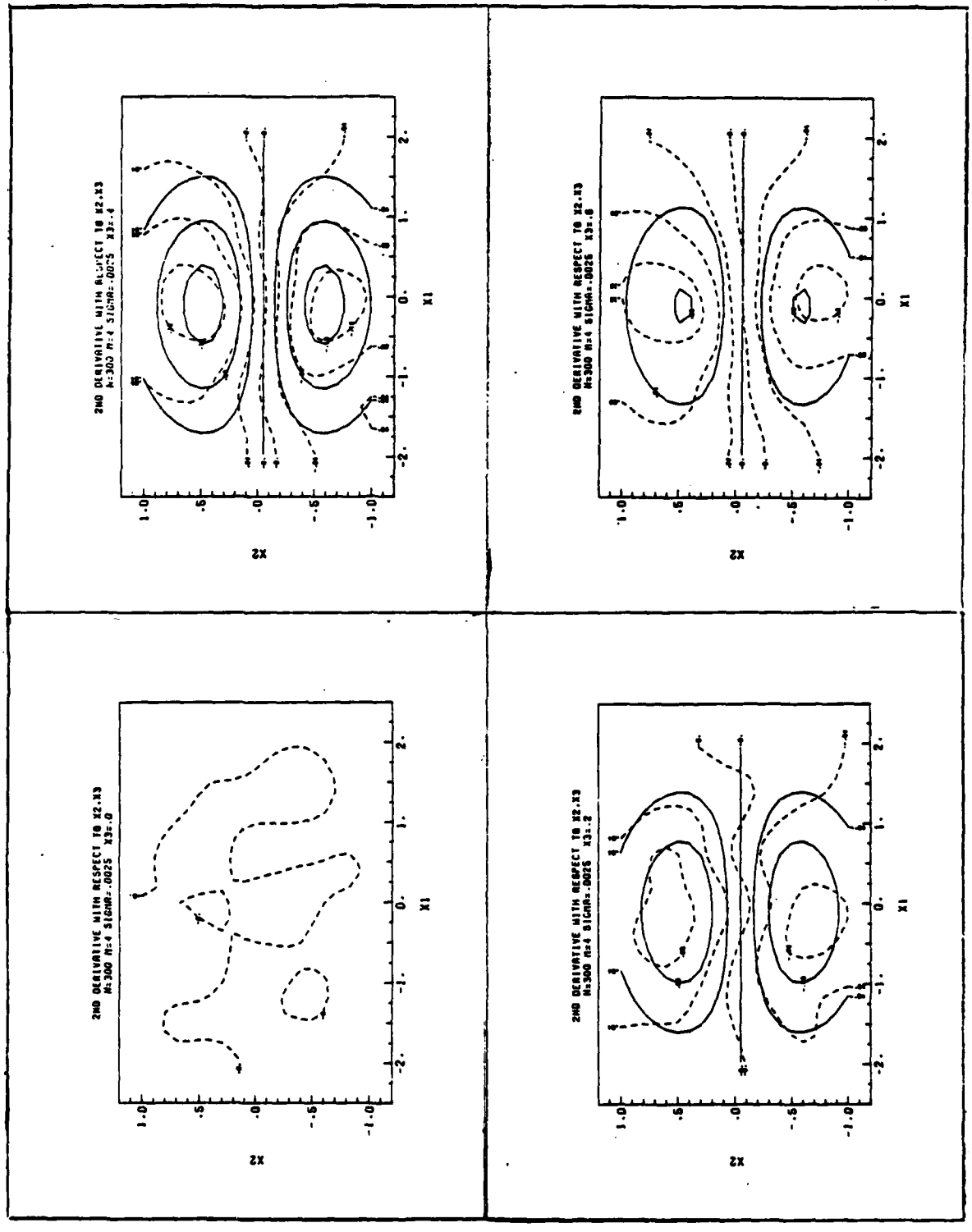


Figure 7.6e: Example 3--solid line is d^2f/dx_2dx_3, dashed line is d^2g/dx_2dx_3.

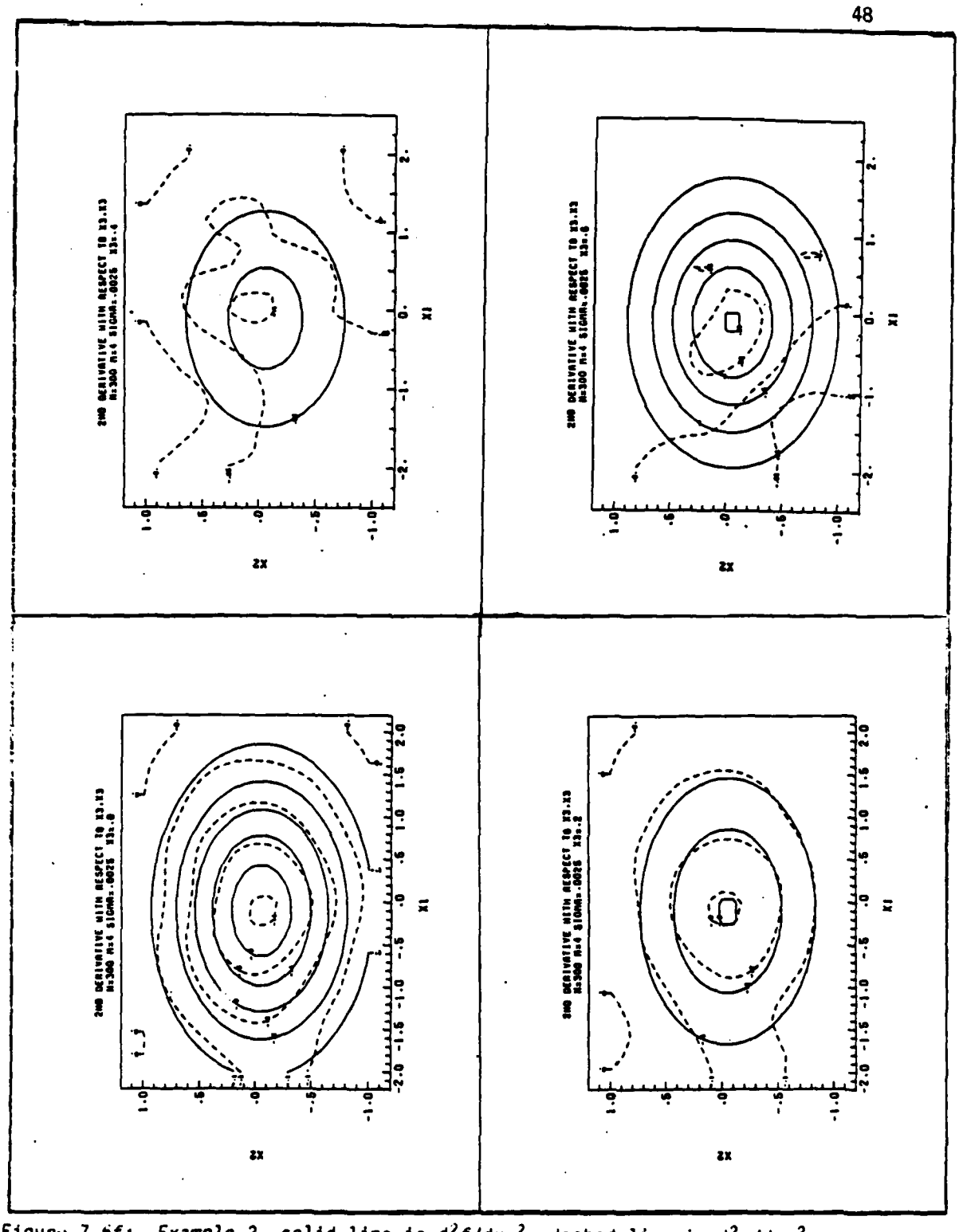


Figure 7.6f: Example 3--solid line is d^2f/dx_3^2, dashed line is d^2g/dx_3^2.

construct data sets $\underset{\sim}{z}_{ks}$, let each element of $\underset{\sim}{z}_{ks}$ equal the corresponding element of $\underset{\sim}{z}$ except for the k^{th}. The k^{th} element is set equal to $f(\underset{\sim}{t}_k) - s\sigma$, σ=.0025. Construct $\underset{\sim}{z}_{ls}$ analogously except that the l^{th} element becomes $f(\underset{\sim}{t}_l) + s\sigma$.

With the data sets $\underset{\sim}{z}_{ks}$ and $\underset{\sim}{z}_{ls}$ probability plots in Figures 7.7 and 7.8 were obtained with MINITAB, Ryan, Joiner and Ryan (1976). The probability plot is constructed by ordering the residuals r_i from smallest to largest and plotting them against their corresponding normal scores. The i^{th} smallest normal score as used by MINITAB is the (i-3/8)/300.25 percentage point of the normal or Gaussian distribution. If the error distribution that is postulated in the model is the correct one, then the probability plot should be nearly linear. In the data sets constructed here the error distribution is not correct because the k^{th} or l^{th} point is biased and contains no random component.

The numbers in Figures 7.7 and 7.8 indicate how many points are plotted at that spot on the graph. An asterisk indicates one point and a plus sign indicates that more than 9 points are overlapping. In Figures 7.7b, c and d the outlier is identified as the point which is separate from the points which form the line. As the assumption of unbiasedness is more strongly violated it shows up more obviously in the plot.

Figures 7.8a-d demonstrate that this outlier detection scheme is not invincible and should be used in conjunction with other diagnostic checks. The point $\underset{\sim}{t}_l$ has very high leverage because it is on the boundary of the data region. In linear regression this is analogous to the points at the extremes

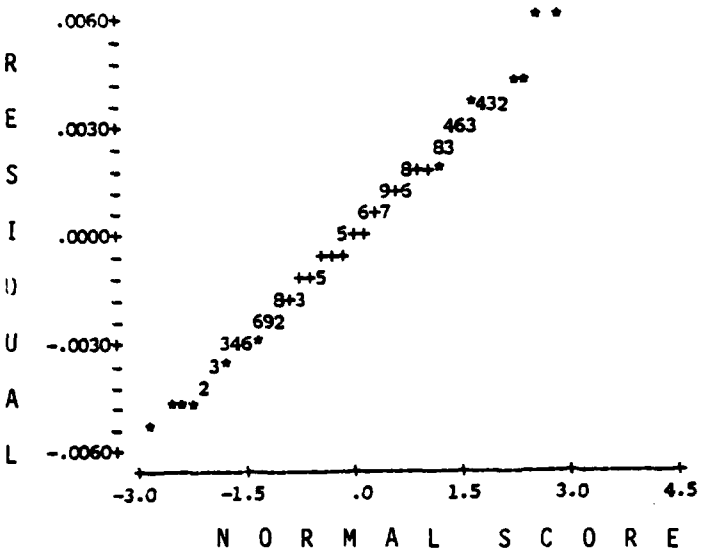


Figure 7.7a: Residuals vs. normal scores for one outlier, $f(t_k) - 0\sigma$, at t_k.

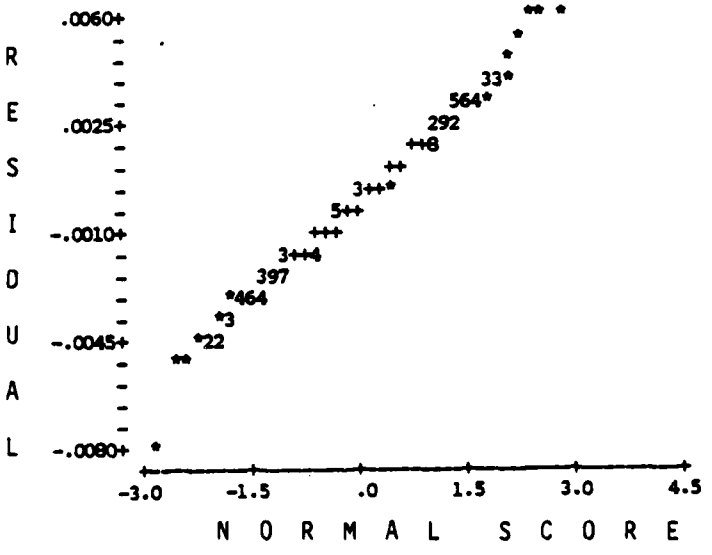


Figure 7.7b: Resdiuals vs. normal scores for one outlier, $f(t_k) - 6\sigma$, at t_k.

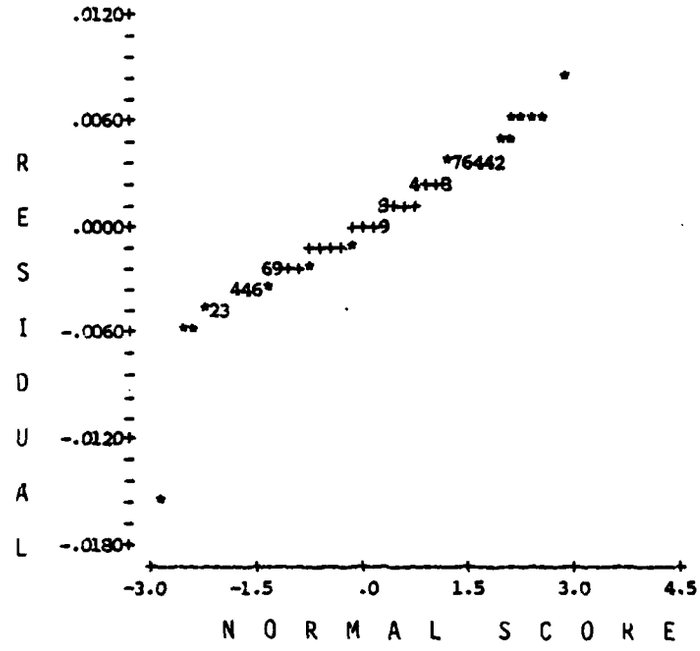


Figure 7.7c: Residuals vs. normal socres for one outlier, $f(\underline{t}_k)$ - 10σ, at $\underline{t}_k$.

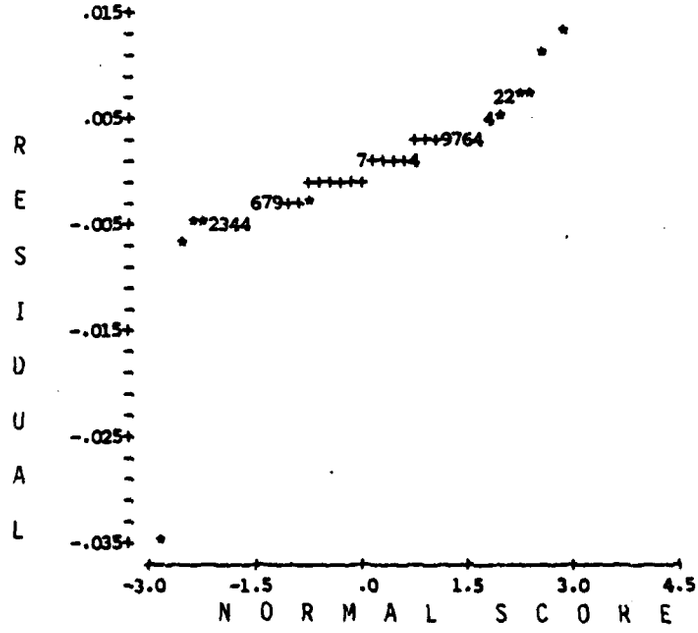


Figure 7.7d: Resdiuals vs. normal scores for one outlier, $f(\underline{t}_k)$ - 20σ, at $\underline{t}_k$.

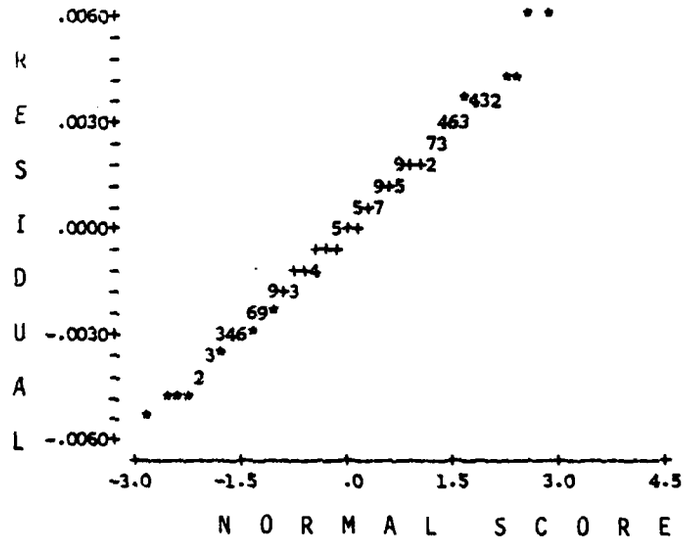


Figure 7.8a: Residuals vs. normal scores for one outlier, $f(\underset{\sim}{t}_1) + 0\sigma$, at $\underset{\sim}{t}_1$.

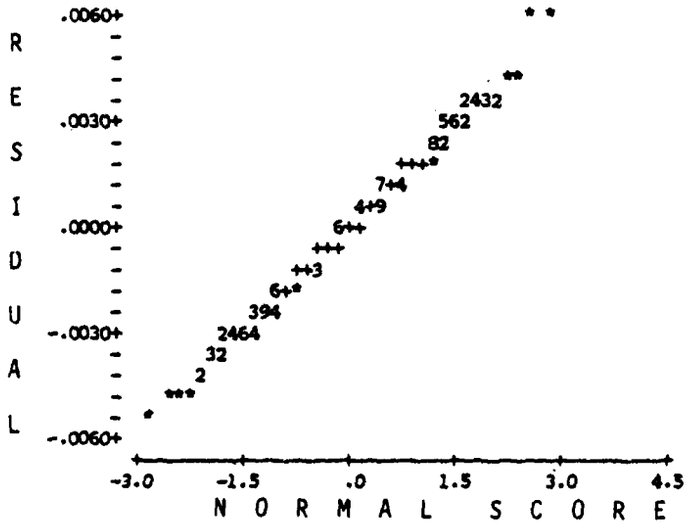


Figure 7.8b: Resdiuals vs. normal scores for one outlier, $f(\underset{\sim}{t}_1) + 6\sigma$, at $\underset{\sim}{t}_1$.

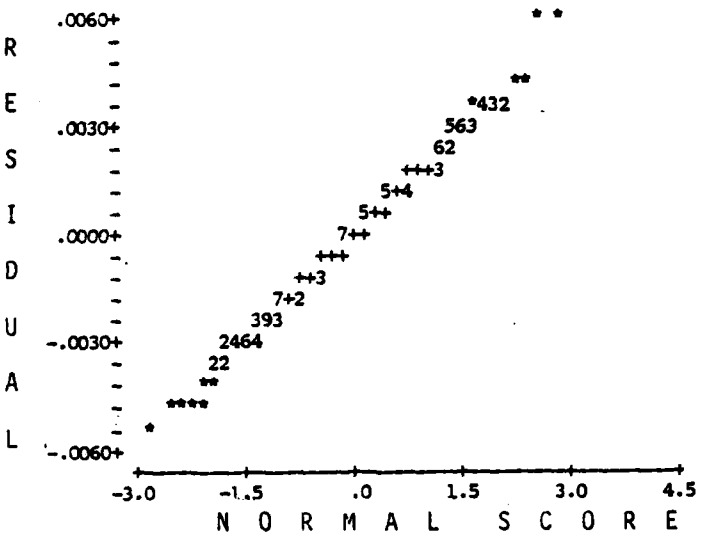


Figure 7.8c: Residuals vs. normal scores for one outlier, $f(t_1) + 10\sigma$, at t_1.

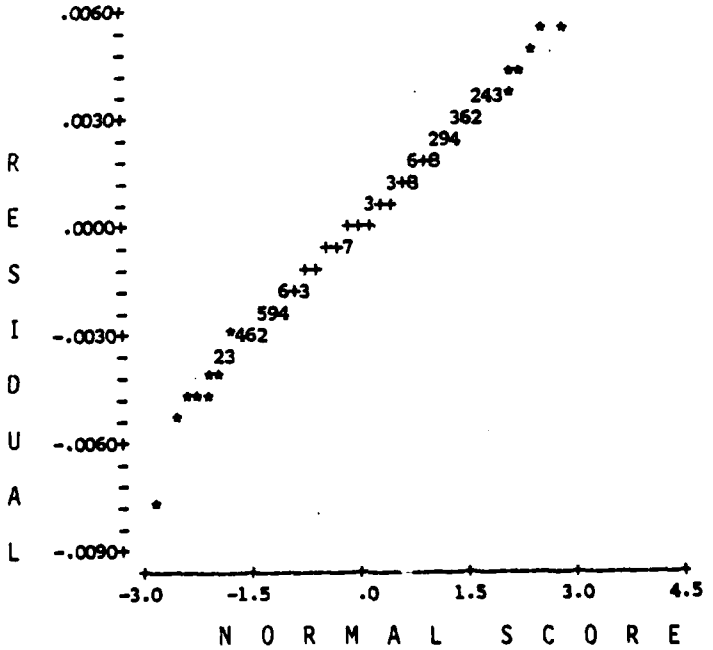


Figure 7.8d: Resdiuals vs. normal scores for one outlier, $f(t_1) + 20\sigma$, at t_1.

of the independent variable range which also have high leverage. Because of this the residual at $\underset{\sim}{t}_1$ is not large and does not show up in the probability plots of Figures 7.8a-d. The leverage at $\underset{\sim}{t}_1$ is so large that it causes another point, the one in the lower left, in Figure 7.8d to appear as descrepant. The probability plot provides a technique to check model assumptions. However, as demonstrated here, this technique should be used in conjunction with other diagnostic checks and with a good understanding of the pitfalls which may be encountered.

Another diagnostic check which may be employed here is to plot the residuals, r_i, against the distance from $\underset{\sim}{t}_i$ to $\underset{\sim}{t}_1$. This is analogous to plotting the residuals against the independent variable in simple linear regression. If a nonrandom pattern is observed, such as serial correlation, then we have evidence that some model assumption is being violated. In practice, $\underset{\sim}{t}_1$ is unknown and hence it may be necessary to do all possible plots, $1=1,\ldots,N$.

If a scaling D_σ had been used then the scaled residuals $D_\sigma^{-1}\underset{\sim}{r}$ would be plotted instead of $\underset{\sim}{r}$.

The procedure described here is a diagnostic method by which some of the model assumptions may be checked. Irregularly spaced multidimensional "noisy" data easily mask outliers. This technique provides a means which may detect these discrepant observations. It is presented here in the hope that it becomes a routine method to check for model violations in an analysis which uses LSS's.

The three dimensional results presented here are new and quite promising. A quantitative measurement of the goodness of fit of the estimated spline and its derivatives to the true function is given in Wendelberger (1981). Further Monte Carlo experiments will be performed in 3 and more dimensions.

8. Running the program

To evaluate an LSS at any point, $\underset{\sim}{t} \in R^d$ involves the execution of two computer programs. The first of these, called MAIN, produces the coefficients of the spline. The second, called EVALUATE, produces the spline, $g_{N,m,\lambda}(\underset{\sim}{t})$. If 2m-2k-d is positive EVALUATE may also be used to produce the first (k=1) or second (k=2) derivative of $g_{N,m,\lambda}$. Depending upon the particular problem at hand the user specifies different options to be exercised by the program. These options will be explained card by card below. Card i will be abbreviated Ci and the commands are summarized in Table 8.1 with an example runstream given in Table 8.3.

C1 is used to specify whether or not the coefficient arrays $\underset{\sim}{c}$ and $\underset{\sim}{d}$ and the matrices X and P used to reconstruct the spline are written to unit 13. X contains the values of the independent variables and P contains the exponents of the polynomials in (2.5), where P is rigorously defined.

To accomplish storing the spline in unit 13 C1 should have SS13 in columns 1 through 4. If EVALUATE is not going to be run then the contents of unit 13 will be unused. In this case C1 should be DONT.

Someone other than the casual user may require other arrays and matrices which are also written to unit 13. See subroutine WRT13 in Wendelberger (1981) for details on the arrays and matrices which are written to unit 13. C2, to be described in the next paragraph, writes into unit 14. See subroutines AWRT14 and BWRT14 to determine the specific values which are written to unit 14.

TABLE 8.1

Input for MAIN

CARD	POSSIBLE VALUES	FORMAT
1	SS13, DONT	A4
2	SM14, UM14, DONT	A4
3	SR15, SP15, VL15, DONT	A4
4	MGCV, USEL	A4
4+	(λ)(Insert if C2 is USEL.)	E15.8
5	VARI, STAN, SAME (Omit if C2 is UM14.)	A4
6	(d,N,m) (Omit if C2 is UM14.)	3I5
7	Format of cards C8+1,...,C8+N.	18A4
8+1	(z_1, $\underset{\sim}{t}_1^T$, σ_1 or σ_1^2)	
.	(z_i) (If C2 is not UM14.)	
.	(z_i, $\underset{\sim}{t}_i^T$, σ_i or σ_i^2) (If C5 is STAN or VARI.)	
.	Format is provided on C7.	(See C7)
8+N	(z_N, $\underset{\sim}{t}_N^T$, σ_N or σ_N^2)	
9	YES, NO	A4

C2 provides the ability to store certain matrices in unit 14 by using SM14 in columns 1 through 4. The storage of these matrices makes it unnecessary to perform the bulk of the computations if a second analysis is to be performed. However, only the dependent variables may be changed for such a subsequent analysis. The relative variances or standard deviations must be identical to the run which used SM14 on C2.

UM14 in the first four columns of C2 provides for use of the matrices which have previously been stored in unit 14. If the value of C2 is DONT then the matrices are neither stored nor used.

C3 provides a means to retrieve certain information during the execution of MAIN and to store this information in unit 15. The first four columns of C3 must be SR15, SP15, VL15 or DONT. If C3 is SR15 the residuals $\underset{\sim}{r} = (\underset{\sim}{z}-\underset{\sim}{g}_{N,m,\lambda}(\underset{\sim}{t}))$ are stored in unit 15 with the format (G24.18). If C3 is SP15 the ordinate and abcissa for each point of the plot of the GCVF as given in the output are stored. First the number (n) of pairs is stored in I5 format followed by the ordered pairs $(i,\ln(V(10^{ai+b})))$, where i is an index number i=1,...,n and ln is the natural logarithm; the format used is (I3,G24.18). If C3 is VL15 then b_i/N, i=1,...,N-M with format (G24.18) followed by $\underset{\sim}{w}$ with with the same format are stored. If none of the above are to be stored then C3 should be DONT.

The value of MGCV on C4 causes the GCVF to be minimized to determine λ^*. If the user wants to supply a value of λ then the value of C4 should be USEL. In that case C4+ is used. C4+ should contain the value of λ in (E15.8) format to be stored in a single precision variable. If C4 is MGCV then C4+ should not be included in the input stream.

C5 is not used if the value of C2 is UM14. Otherwise C5 is used to input relative variances or relative standard deviations or neither of these for the errors of the dependent variable. If the relative variances are to be read then C5 should be VARI; if the relative standard deviations are read then C5 is STAN; and if neither is read then C5 is SAME. The value SAME is equivalent to that of entering all 1's as the relative variances. However, if SAME is used then the program circumvents both multiplication and division by 1 since D_σ is simply the identity matrix.

C6 is not used if C2 is UM14. Otherwise C6 reads in the number of independent variables (= dimension), the number of observations N and the value of m to be used. The format used is (3I5).

C7 contains the format to be used to read in the data values. The format should require at most 72 spaces including the left- and right-most parentheses.

The data follow in C8+1 through C8+N. The data should be real Fortran variables, each data line should contain, in order, the dependent variable, the independent variable(s) and the relative variance or standard deviation if used. If C2 is UM14 then C8+1 through C8+N should contain only the dependent variables. They should be given in the identical sequence as the dependent and independent variable(s) were when C2 had the value SM14.

The last card to be read is C9. It should contain one of the values YES or NO . If YES then experimental confidence intervals are provided along with degrees of freedom and an estimate of the variance (Wahba, (1981)). If NO then these values are neither computed nor printed.

To evaluate the spline (k = 0), first derivative (k = 1) or second derivative (k = 2) the program EVALUATE is used. Previous to running EVALUATE the program MAIN must have been run with C1 writing the coefficients to unit 13 (C1 must have been SS13). EVALUATE will then read the matrices from unit 13 and calculate the spline, its first derivative or second derivative. The kth derivative (k = 1 or k = 2) will be calculated only if 2m-2k-d is greater than 0. A description of the input stream for EVALUATE is given in Table 8.2 with a sample runstream given in Table 8.3.

C1 contains two integer values in (2I5) format. The first integer, N', specifies the number of points $\underset{\sim}{t} \in R^d$ at which the function is to be evaluated. The second integer should be one of 0, 1 or 2 depending upon whether the spline, first or second derivative, respectively, is to be calculated.

The second card contains the format to be used to read in the N' points. The format should require at most 72 spaces, including the left- and right-most parentheses. The independent variables are read line by line in the same sequence as that which was used to calculate the coefficients.

C3 must be either SV15 or DONT. To store the values in unit 15, C3 should be SV15. This causes the values followed by the corresponding independent variable(s) to be written to unit 15. If C3 is DONT then the values are not written to unit 15.

C3+ is used only if C3 is SV15. Then C3+ should have the format which is to be used to write the calculated value(s) followed by the independent variable(s) into unit 15. This format may have at most 72 spaces including both the left- and right-most parentheses.

TABLE 8.2

Input for EVALUATION

CARD	POSSIBLE VALUES	FORMAT
1	(N',k)	2I5
2	Format to read C4+1,...,C4+N'.	18A4
3	SV15,DONT	A4
3+	Format for 15 (Omit if C2 is DONT.)	18A4
4+1		
.	Independent variable points	
.	of evaluation, $\underline{t}^T$.	(See C2)
.	Format is provided in C2.	
4+N'		

TABLE 8.3

Sample Runstreams	Comments
@XQT SMOOTH*SPLINE.MAIN	Implements the MAIN program.
SS13	Stores the spline coefficients in unit 13.
SM14	Stores matrices in unit 14.
DONT	Doesn't store other values.
MGCV	Minimize the GCVF to determine λ^*.
SAME	The relative variances are all the same.
1 24 2	One dimension, 24 observations, m=2.
(F3.10,33X,F4.0)	Format of the input data.
@ADD DATA.	Inserts data from Table 3.1 in runstream.
YES	Provide confidence intervals.
@XQT SMOOTH*SPLINE.EVALUATE	Implements the EVALUATION program.
200 0	At 200 points evaluate the spline.
(36X,F8.4)	Format of the independent variables.
SV15	Store the spline and independent variable values in unit 15.
(2E15.8)	Format of above.
@ADD PLOTDATA.	Inserts abcissa points to be used for plotting.
@XQT SMOOTH*SPLINE.MAIN	Implements the MAIN program.
SS13	Stores the spline coefficients in unit 13.
UM14	Uses the matrices stored in 14 by MAIN above.
DONT	Doesn't store other values.
USEL	Use the following value of λ.
.00016E00	Value of λ to be used.
(F3.0)	Format of the dependent variables.
@ADD DATA.	Inserts data from Table 3.1.
YES	Provides confidence intervals.
@XQT SMOOTH*SPLINE.EVALUATE	Implements the evaluation program.
200 0	At 200 points evaluate the spline.
(36X,F8.4)	Format of the independent variable.
SV15	Store the spline and independent variable in 15.
(2E15.8)	Format of above.
@ADD PLOTDATA.	Inserts abcissa points to be used for plotting.

C4+1 through C4+N' contain the independent variable(s) at which the function is to be evaluated. These should be in the format given on C2. The independent variable(s) should be in the same sequence as used to obtain the coefficients with the program MAIN.

The programs MAIN and EVALUATE are written in ASCII FORTRAN Level 9R1 and are running on the UNIVAC 1100/80 computer at the University of Wisconsin. All calculations are performed in double precision.

The subroutines used by the programs MAIN and EVALUATE are named: AWRT14, BWRT14, CALC, CALD, CALRES, CHECKQ, COLOFK, CONINT, DATAR, DERIV1, DERIV2, E, ED1, ED2, GETASI, GETBM, GETR, GETRDE, GETTHM, GRAPHV, MAKEB, MAKETS, MINVL1, MINVL2, MQRDC, PRINT, PRNTLM, RCHECK, READ13, SPLINE, SVDB, VARDF, VLHELP, VOFL, WHATDO, WRT13, AND WRT15. GRAPHV, MINVL1 and MINVL2 are modeled after similar subroutines of the one dimensional smoothing spline program written by Fleisher (1979) and running at the Madison Academic Computing Center (MACC). A description of the program structure is given in Wendelberger (1981).

The following LINPACK subroutines are also used by the program MAIN: DAXPY, DCOPY, DDOT, DNRM2, DQRDC, DQRSL, DROT, DROTG, DSCAL, DSVDC, DSWAP and DTRSL. The code for these routines is not included here. It may be found in the LINPACK USERS' GUIDE by Dongarra, Bunch, Moler and Stewart (1979). One modification is made in the LINPACK subroutine DSVDC: the parameter MAXIT is increased from 30 to 60. This parameter sets the maximum number of iterations to be performed in the algorithm to determine the singular values and vectors of B before termination due to nonconvergence. Increasing MAXIT to 60 is

necessary because with large N, say N > 140, 30 iterations may not be large enough for some problems. An example with N=150 failed because MAXIT=30 was too small. However, with MAXIT=60 example 3 with N=300 was successfully run. In fact MAXIT=60 has proved ample for all examples tried to date. The version of the program described here uses the singular value decomposition to obtain the spectral decomposition of B. A new modified version uses the EISPACK (Smith, et al., (1976)) routines DTRED2 and DTQL2 to accomplish this task at a much reduced cost and at no loss in accuracy. This is because the singular value decomposition does not make use of the symmetry of B. The EISPACK routines do make use of the symmetry of B and thus the cost of the decomposition is roughly cut in half.

ACKNOWLEDGMENTS

The author wishes to express thanks to Grace Wahba for introducing this problem and providing guidance and encouragement, to Gene Golub who inspired this algorithm and to the attendees of both the First and Second SIAM Summer Research Conference on Numerical and Statistical Analysis from whose comments this work has benefited.

Thanks go to Alison Pollack for her comments on earlier drafts of this report and to the students of Statistics 840, fall semester 1980, who endured the earlier versions of the program.

This research was sponsored by the National Aeronautics and Space Administration under NASA Grant No. NAG5-128 and by the Office of Naval Research under Contract No. N00014-77-C-0675.

REFERENCES

Chatterjee, S. and Price, B., 1977: Regression analysis by example. John Wiley and Sons, 10-18.

Craven, P. and G. Wahba, 1979: Smoothing noisy data with spline functions: Estimating the correct degree of smoothing by the method of generalized cross-validation. Numer. Math., 31, 377-403.

Dongarra, J. J., J. R. Bunch, C. B. Moler, and G. W. Stewart, 1979: Linpack users' guide. Society for Industrial and Applied Mathematics, Philadelphia, Pennsylvania.

Duchon, Jean, 1976: Interpolation des fonctions de deux variables suivant le principe de la flexion des plaques minces. R.A.I.O. Analyse numerique, Vol. 10, No. 12, 5-12.

Fleisher, J., 1979: Spline smoothing routines. Reference manual for the 1110. Academic Computing Center, University of Wisconsin, Madison.

Franke, R., 1979: A critical comparison of some methods for interpolation of scattered data, Naval Postgraduate School Report No. NPS-53-79=003, March, 1979.

Golub, G. H. and C. Reinch, 1970: Singular value decomposition and least squares solutions. Numer. Math. 14, 403-420.

Harder, R. L., and R. N. Desmarais, 1972: Interpolation using surface splines. J. Aircraft, Vol. 9, No. 2, 189-191.

Lucas, H. A., 1978: Estimation of smoothing parameters to smooth noisy data and confidence regions for the underlying function. Ph.D. thesis, Dept. of Statistics, University of Wisconsin, 136 pp.

MACC, 1978: Random Number Routines. Reference Manual for the 1110. Madison Academic Computing Center, University of Wisconsin-Madison.

Mosteller, F. and J. W. Tukey, 1968: Data analysis including statistics, Handbook of Social Psychology, Vol. 2, Reading, MA, Addison-Wesley, 80-203.

Prenter, P., 1975: Splines and variational methods, John Wiley and Sons, Inc.

Reinsch, G., 1967: Smoothing by spline functions. Numer. Math., 10, 177-183.

Ryan, T. A., B. L. Joiner and B. F. Ryan, 1976: MINITAB student handbook, Duxbury Press.

Schoenberg, I. J., 1964: Spline functions and the problem of graduation. Proc. of Nat. Acad. of Sciences. 52, No. 4, 947-950.

Smith, B.T., J. M. Boyle, J. J. Dongarra, B. S. Garbow, Y. Ikebe, V. C. Klema, and C. B. Moler, 1976: Matrix Eigensystem Routines--EISPACK Guide, Vol. 6. Lecture notes in Computer Science, Springer-Verlag, 551 pp.

Wahba, G., 1979: How to smooth curves and surfaces with splines and cross-validation, Tech Report #555, Dept. of Statistics, University of Wisconsin.

Wahba, G., 1980: Ill posed problems: Numerical and statistical methods for mildly, moderately and severely ill posed problems with noisy data. Tech Report #595, Dept. of Statistics, University of Wisconsin.

Wahba, G., 1981: Bayesian confidence intervals for the cross validated smoothing spline. Tech. Report No. 645, Dept. of Statistics, University of Wisconsin.

Wahba, G. and J. Wendelberger, 1980: Some new mathematical methods for variational objective analysis using splines and cross validation. Mon. Wea. Rev., Vol. 108, 8, 1122-1143.

Wahba, G. and S. Wold, 1975: A completely automatic French curve: Fitting spline functions by cross-validation. Communications in Statistics, 4, 1-17.

Wendelberger, J. G., 1981: Smoothing noisy data using multivariate splines and generalized cross-validation. Ph.D. thesis to appear, Dept. of Statistics, University of Wisconsin.

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER Technical Report No. 648	2. GOVT ACCESSION NO. AD-A114 746	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) THE COMPUTATION OF LAPLACIAN SMOOTHING SPLINES WITH EXAMPLES		5. TYPE OF REPORT & PERIOD COVERED Scientific Interim 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) James Wendelberger		8. CONTRACT OR GRANT NUMBER(s) No. N00014-77-C-0675
9. PERFORMING ORGANIZATION NAME AND ADDRESS Department of Statistics, Univ. of Wisconsin 1210 W. Dayton St. Madison, WI 53706		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Office of Naval Research 800 Quincy St. Arlington, VA		12. REPORT DATE September 1981 13. NUMBER OF PAGES 68
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Distribution of this report is unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Spline smoothing; Computation of splines; multivariate derivative estimation; Spline confidence intervals; Spline diagnostic checks.

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

Laplacian Smoothing Splines (LSS) are presented as generalizations of graduation, cubic and thin plate splines. The method of generalized cross validation (GCV) to choose the smoothing parameter is described. GCV is used in the algorithm for the computation of LSS's. An outline of a computer program which implements this algorithm is presented along with a description of the use of the program. Examples in one, two and three dimensions demonstrate how to obtain estimates of function values with confidence intervals and estimates of first and second derivatives. Probability plots are used as a diagnostic tool to check for model inadequacy.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-LF-014-6601

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)